Chapter 1

MACHINE TRANSLATION

Machine translation is one of the holy grails of natural language processing. It is a seemingly well-defined task: converting text in one language into another while preserving its meaning. It mirrors a human activity that is done by amateur bilingual speakers and professionals on a daily basis. But at the same time, it is fraught with difficulties so that most researchers do not expect to reach human quality anytime soon. The goal is more modest: producing translations that are *good enough*, or *useful*.

In recent years, with the advent of the world wide web and the emergence of data-driven methods, machine translation research has received a jolt of new activity and much more visibility. An expanding number of research groups have taken on the problem. Everybody has access to machine translation by visiting popular web sites such as Google Translate or Systran's Babelfish.[1]

The prominence of machine translation research signifies two things: Machine translation systems have matured to the point that they offer a useful service for a large number of people. But the visible lack of accuracy also shows that much work still needs to be done, maybe not to reach perfect translation, but reach ever higher levels of quality.

1.1 Machine translation today

The most profound recent change in the long history of machine translation can be dated back to 1988. While machine translation research started in the 1940s, a group of researchers at IBM proposed to radically change the approach to machine translation. *A statistical approach to language translation* [Brown et al., 1988] would not anymore require a large group of linguists toiling away at defining the transformations and lexicons that powered traditional systems up to that point. Instead, large corpora of

[1] `http://translate.google.com/` and `http://babelfish.yahoo.com/`

translated texts, typically many millions of words, would provide the answer and a clever statistical model would help to learn the rules of translation and provide the basis for a decoding algorithm that finds the best translation for a given input sentence.

The rather simple models proposed by IBM at that time (we will discuss them in some detail in the Section 1.3 on page 9 on word alignment) have evolved over the last two decades into so-called phrase-based model (Section 1.4 on page 16) and tree-based models (Section 1.5 on page 25).

Current research pursues several directions, most notably

- the development of models that mirror more closely linguistic understanding of language,
- the application of novel machine learning methods to the estimation problem of learning translation rules from the data, and
- the attempts to exploit various types of data sources, which are often not in the desired domain or may not be even proper sentence-by-sentence translations at all.

Machine translation is being integrated into various applications: cross-lingual information retrieval, speech translation, tools for translators, to name a few.

This chapter focuses on the basic methods that underlay modern machine translation systems. But before that, let us make first sure that we know what our goal is and how we evaluate that we are coming closer.

1.2 Machine translation evaluation

We defined machine translation above as *converting text in one language into another while preserving its meaning* — and there it is, the word **meaning**. While it may inspire the minds of philosophers, it is a dreaded word for the engineer. What is meaning? How can we measure it? How do we know that two words, phrases, or sentences have the same meaning? And if they are close, how close?

The fact that there are almost as many metrics for machine translation as there are research groups working in the field [Callison-Burch et al., 2008], [Callison-Burch et al., 2009], [Przybocki et al., 2008] is testament to how this indeed not a trivial problem. Figure 1.1 on the next page shows what happens when we ask ten different human translators to translate one sentence,

这个 机场 的 安全 工作 由 以色列 方面 负责 .
Israeli officials are responsible for airport security.
Israel is in charge of the security at this airport.
The security work for this airport is the responsibility of the Israel government.
Israeli side was in charge of the security of this airport.
Israel is responsible for the airport's security.
Israel is responsible for safety work at this airport.
Israel presides over the security of the airport.
Israel took charge of the airport security.
The safety of this airport is taken charge of by Israel.
This airport's security is the responsibility of the Israeli security officials.

Figure 1.1. Ten different translations of the same Chinese sentence, created by different human translators (a typical example from the 2001 NIST evaluation set)

here from Chinese to English. Even for such a short sentence, each translator comes up with a different translation. This is not due to the nature of Chinese — the same can be demonstrated for other languages.

This means that if we translate a Chinese sentence into a machine translation system, the result would very unlikely match a human translation, even if it were a perfectly fine translation. So, how do we know whether it is a correct translation?

Since we cannot expect that it matches one of the references, we need to have some measure that matches the meaning of the system output against the meaning of the source, or as it is more commonly done, against the meaning of human reference translations.

1.2.1 Human assessment

Since we do not trust computers to deal with problems of meaning, a common retreat is to assign the task to human judges. Given the source and the system output, we may ask them if the output constitutes a correct translation.

Figure 1.2 on the following page shows how four different human judges assessed translations of a French sentence into English in a recent study [Koehn and Haddow, 2009]. While the judges agreed for some translations, for most there is disagreement. So, even a simple task such the assessment of correctness of translation does not lead to clear answers.

Is this a problem? No. In the probabilistic view of the world that we adopt in statistical machine translation, there are no clear answers. Some answers are just more likely than others. For each translation there is a

correct	Sans se démonter, il s'est montré concis et précis.
1/3	Without fail, he has been concise and accurate.
4/0	Without getting flustered, he showed himself to be concise and precise.
4/0	Without falling apart, he has shown himself to be concise and accurate.
1/3	Unswayable, he has shown himself to be concise and to the point.
0/4	Without showing off, he showed himself to be concise and precise.
1/3	Without dismantling himself, he presented himself consistent and precise.
2/2	He showed himself concise and precise.
3/1	Nothing daunted, he has been concise and accurate.
3/1	Without losing face, he remained focused and specific.
3/1	Without becoming flustered, he showed himself concise and precise.

Figure 1.2. Human judgments on translations. Four different human evaluators often disagree if a translation is correct, for instance the first translation of the French sentence was judged as correct by one judge and wrong by three others.

Reference:	Israeli officials are responsible for airport security.
System A:	Israeli officials are responsible for security.
System B:	Israeli officials are responsible for rail security.
System C:	Israeli officials are not responsible for airport security.
System D:	Israeli officials are responsible. For airport security.
System E:	Israeli officials are responsible for arport sequrity.

Figure 1.3. Five different translation with mistakes. How would you rank them?

probability distribution over possible judgments. If we have enough samples, our statistics converge to the true distribution, and hence valid assessments. In the world of meaning, there is no true black and white. There will always be someone who finds fault with a translation.

In practice, machine translation systems will produce translations that will have some mistakes. Especially for long sentences of, say, 30 words, we cannot expect flawless output. Moreover, we are often not interested in absolute assessments (*How many sentences are translated correctly?*) but in comparisons of systems (*Is system A better than system B?*). So, instead of asking if a translation is correct, we more often are in a situation where we ask if one translation is better than another.

See Figure 1.3 for an artificial example of five different systems that produce five different translations, each of them with a different mistake, be it a missing word, mistranslation of a word, insertion of the word *not*, wrong punctuation, or spelling errors. Which translation do you prefer?

Again, not a straightforward task. Human judges have different preferences. Some may be very obsessed with punctuation [Truss, 2003], while

others could not care less. How bad is the insertion of a simple function word? What if it is *not*?

We may want to break up the simple question of **correctness** into more fine-grained distinctions. Is the translation **fluent**, i.e., is the output well-formed in the target language? Is the translation **adequate**, i.e., is the meaning preserved regardless of output language quality? Even with such categories different human judges may have different preferences.

And here we are getting a bit more concerned. We are setting up a very artificial task for a human judge. Nobody except for language teachers who are grading exams would look at translations and assess their quality in isolation. Humans use translations to fill an information need. If the translation of a foreign text gives them the answers they were looking for, then it was successful.

To truly test machine translation quality, we need to place it in a setting that involves its use. There have been some recent efforts to create **task-based** evaluation methods. For instance, we may give a human assessor a translated text and then ask her questions about the content. If she able to answer them, the translation was successful [Jones et al., 2006]. In a variation of this method, we may ask a human assessors to edit the translation to produce fluent output without access to the source. Then, we check if the edited translation is correct, hence testing her understanding [Callison-Burch et al., 2009].

1.2.2 Automatic evaluation metrics

The development of machine translation systems requires frequent evaluation; too frequent for costly manual evaluation. One of the turning points of statistical machine translation research was the establishment of a regime of automatic evaluation that is commonly accepted. In fact, papers on improvements in machine translation rarely include human evaluation, but they pretty much have to include improvements in the current most popular automatic evaluation metric BLEU.

How can we trust a computer program that computes a metric score to give us reliable assessment of machine translation quality? If a computer program could tell us if a translation is correct or wrong, why could it not produce a correct translation in the first place? Well, it uses one trick and one cop-out.

The trick is to not only use the source and system output, but also to use one or more reference translations that were produced by reliable human translators. We have already discussed at length that human and machine

translation may lead to translations that are different from existing reference translations but still be correct. But, and here is where the cop-out comes in and the argument gets a bit murky: if the machine translation is similar to existing reference translations, then it is more likely to be correct. While it is easy to defeat this argument with a single sentence example, the case for automatic metrics is built on the use of a large test set of 100s or even 1000s of sentences. Over a large test set, better translations are expected to be more similar to references.

Developers of machine translation evaluation metrics do not just appeal to such an argument, they validate this claim by carrying out correlation studies that show that their metrics rank systems in pretty much the same way as human judges would do. There are even evaluation campaigns for evaluation metrics, where different metric developers compete for the highest correlation with human judges [Callison-Burch et al., 2008], [Callison-Burch et al., 2009], [Przybocki et al., 2008].

We have now established a regime to test performance during development of a machine translation system: We first select a test set. We give it to human translators to produce one or more translations. We then run our machine translation system, and measure how similar the output is to the reference translations. Then, we make a change to our machine translation system, run it again over the same test data, measure similarity again and see if it has improved.

We are left with the task to define a measure of **similarity** between machine translation output and reference translations. This is again one of these dreaded words that are in the same sphere as *meaning*, but we will start simply.

1.2.3 WER, BLEU, METEOR, ...

Language is made up out of words, so two sentences are similar, if they share a lot of words. So, when comparing machine translation output and a reference translation, we can count: (a) **matches**, words that are both in the reference and in the output, (b) **insertions**, words that are only in the output, and (c) **deletions**, words that are only in the reference.

Given these three statistics, we can compute a number of metrics:

$$\text{precision} = \frac{\text{matches}}{\text{matches+insertions}} \tag{1.1}$$

$$\text{recall} = \frac{\text{matches}}{\text{matches+deletions}} \tag{1.2}$$

$$\text{PER} = 1 - \frac{\text{matches}}{\text{matches+max(insertions,deletions)}} \tag{1.3}$$

$$\text{f-score} = \frac{2\times\text{precision}\times\text{recall}}{\text{precision+recall}} \tag{1.4}$$

$$\text{weighted f-score} = \frac{(1+\alpha)\times\text{precision}\times\text{recall}}{\alpha\times\text{precision+recall}} \tag{1.5}$$

All these metrics are the basis of machine translation metrics that have been proposed over the last years. There is some debate, if precision or recall is more important, which is related to the question how to penalize too short and too long translations. PER, **position-independent errror rate**, is one of the earliest metrics that have been proposed.

Let us now look at a number of refinements that have been applied to this simple idea of matching words between machine translation output and reference translations.

The first refinement is the use of **multiple reference translations**. Given the allowable variation in translation, it may be too constraining to use only a single reference translation as the gold standard. If we have multiple translations, the chances that correct machine translation output matches one of them very closely increases. This should reduce the problem of correct but badly scoring translations. In terms of integrating multiple reference translations into a metric, we may always choose the best score with any of the reference translations, or come up with more elaborate schemes. For instance, we may count as a match if an output word matches in any of the reference translations.

Secondly, we may match not only words, but also **n-grams** of words. This is an attempt to take word order into account. We may not expect that all the matching words in output and reference are in the same order, but if neighboring words in the output match neighboring words in the reference translation, then this is certainly a plus.

These two refinements are the basis of the BLEU score [Papineni et al., 2002]. It is the most commonly used metric in the field, so it is worth taking a closer look. The formal definition is:

$$\begin{aligned} \text{BLEU} &= \text{brevity-penalty} \times \exp\left(\frac{1}{4}\sum_{i=1}^{4}\log \text{precision}_i\right) \\ \text{brevity-penalty} &= \min\left(1, \frac{\text{output-length}}{\text{reference-length}}\right) \end{aligned} \tag{1.6}$$

BLEU is in essence the geometric mean of n-gram precisions, typically using n-grams of length 1–4 (precision_i is precision for n-grams of size i). Since it is a precision-based metric, there is a need to hedge against too short translations. This is done with a brevity penalty, which only comes into effect if the output length is shorter than the reference length. Multiple reference translations are incorporated by allowing n-gram matches against any of the reference translations. If an n-gram occurs multiple times in the output, it is only counted as correct as many times as the maximum number of times it occurs in one of the reference translations. The reference length for multiple references is chosen by selecting the closest length to the output length.

The BLEU score is computed over entire documents or test sets, not for single sentences. In fact, it is not a very good metric for single sentences, since the 4-gram precision is often zero, or otherwise has too strong of an impact. When using BLEU on the sentence level, the precision is typically smoothed by adding 1 to the number of actual matches.

Since the BLEU score was proposed in 2002, a number of further refinements have been proposed. One is to relax the restriction of matches to exact surface form matches. We may also want to give at least partial credit for words that only differ in their morphological variation but derive from the same **lemma**. We may also give credit to **synonym** matches utilizing resources such as WordNet [Miller et al., 1993]. A metric that recently gained some prominence, METEOR [Banerjee and Lavie, 2005], allows for such matches and also relies more heavily on recall instead of precision.

An old idea is to not only treat sentences as bag of words or n-grams, but to compute a explicit word alignment between the machine translation output and the reference translation. A metric borrowed from speech recognition, word error rate (WER), enforces such an alignment and does not allow for any reordering between the sentences. Since there is a significant amount of allowable variation in word order without change in meaning, WER has been refined into metrics that do allow movements but penalize them as additional errors (similar to insertions and deletions). TER, which is known as translation error rate or translation edit rate, computes the minimal cost alignment between output and reference allowing for moves. Unfortunately, finding the optimal alignment is a computationally hard problem, so this metric is slow in practice and usually an approximation is computed.

Finally, we have all the ingredients for setting up machine translation evaluation as a machine learning problem. Over the years, evaluation campaigns have created training data in terms if system translations and their human assessment. We have a well-defined goal: optimizing correlation of

an automatic metric with the human judgments. So, we can use any kind of features in a machine learning approach. In recent years, researchers have incorporated linguistic features such as syntactic relationships or semantic roles.

1.3 Word alignment

The idea of statistical machine translation is to learn translation rules from a sentence-aligned parallel corpus. Let us start with the extraction of word translations from such a corpus. Detecting word translations provides the basis for establishing a word alignment, a fundamental step in any statistical machine translation model.

1.3.1 Co-occurence

We assume that we have a sentence-aligned parallel corpus, where each foreign sentence **f** is paired with its English translation **e**. Such corpora are available on the Internet (for instance Europarl[2] or from the LDC[3]), or are collected by translation agencies who call them translation memories. Raw corpora require some basic pre-processing, typically tokenization (separating out punctuation), data cleaning (throwing out very long sentences or sentences mismatched due to their relative length), and normalizing upper and lower case (for instance, by lowercasing all words), and we are ready to go.

We would like to learn which words in one language translate into words of the other language. We want to learn this in form of a probability distribution $t(e|f)$ that gives for each foreign word f and each English word e the probability that e is a translation of f. For instance, for the German word *Haus* we expect to learn something like this:

$$t(e|Haus) = \begin{cases} 0.8 & \text{if } e = \textit{house}, \\ 0.16 & \text{if } e = \textit{building}, \\ 0.02 & \text{if } e = \textit{home}, \\ 0.015 & \text{if } e = \textit{household}, \\ 0.005 & \text{if } e = \textit{shell}. \end{cases} \tag{1.7}$$

As the name statistical machine translation implies, we learn such a model from the statistics of the data, i.e., the counts that we see in our

[2] http://www.statmt.org/europarl/
[3] http://www.ldc.upenn.edu/

parallel corpus. We can go over all sentence pairs that contain a specific foreign word f (like *Haus*), and see what English words occur on the English side. Based on these counts, we can estimate the conditional probability distribution:

$$\widehat{t}(e|f) = \frac{\text{count}(f, e)}{\sum_{e'} \text{count}(f, e')} \tag{1.8}$$

We have to be a bit careful with the counting. Let us say, we have foreign sentence $\mathbf{f}$ that contains the German word f in question. On the English side, there are five words. Can we now treat each co-occurence of f with each of the five words $e \in \mathbf{e}$ as one count?

We could, but this would lead to different count collections for short and long sentences. In a sentence pair with five English words, we would collect five counts for the foreign word f. But if there are ten English words, we would collect ten counts. But in fact, the foreign word f occurs in each instance only once.

So, instead, we are resorting to fractional counting: If there are five English words, and since we do not know which of them is the translation of f, we count each word as $\frac{1}{5}$.

How well will this work? Intuitively, an English word e that is a common translation of f will co-occur frequently with that word, so we would expect to estimate a relatively high $\widehat{t}(e|f)$. But, then, the English period at the end of the sentence will occur in nearly every English sentence, so it will likely co-occur more frequently with f than any of its true translations individually. But is the most likely translation of every foreign word really the period?

Something has gone awry. We normalize the co-occurence counts (f, e) with the frequency of f, but not with the frequency of e. There are now several other statistical measures we could use to do our estimation. For instance, we could look at mutual information. In fact, several such measures have been used in the literature to massage co-occurence statistics.

1.3.2 IBM Model 1

The very first model of machine translation, IBM Model 1, tackles the estimation problem in a different way. Instead of changing the conditional probability model, it forces us to find for each sentence pair a word alignment. The probability of a foreign sentence $\mathbf{f}$ to translate into an English sentence $\mathbf{e}$ with a alignment a is defined as:

$$p(\mathbf{e}, a|\mathbf{f}) = \frac{1}{Z} \prod_{j=1}^{l_e} t(e_j|f_{a(j)}) \tag{1.9}$$

The formula includes the alignment function a which is straightforward to explain: it matches each English position j to a foreign word position $a(j)$. Note that the formula above is a slight simplification of the original IBM Model 1. We avoid the introduction of the noisy channel model[4] at this point. We are also glossing over the normalization Z that ensures that $p(\mathbf{e}, a|\mathbf{f})$ is a proper probability distribution.

Let us say that we went through the estimation process that we did in the section above. We ended up with a conditional distribution $t(e|f)$. We now look at the first sentence pair of the parallel corpus again, and would like to find the most likely word alignment.

A closer look at Equation 1.10 on page 13 reveals that maximizing $p(\mathbf{e}, a|\mathbf{f})$ means that we independently maximize each $t(e_j|f_{a(j)})$. In other words, we need to find the one foreign word f in the sentence $\mathbf{f}$ that best explains e.

You may still have in mind the mess we got ourselves into in the previous section. Each foreign word f would prefer to produce the English period. But this is not the question we are asking here. Only one foreign word gets to produce the period. But we also need to explain the other English words. What is the foreign word that best explains an occurrence of the English *house*? It is certainly not the foreign period, which is a very confused word that spreads its probability mass over very many words. We would expect that the German *Haus* has a fighting chance to be chosen to align to *house*. Yes, it is also confused and may have an odd preference for producing periods, but we would expect $p(house|Haus)$ to be bigger than $p(house|.)$.

Let us go one step further: if we go through each sentence pair of the parallel corpus, find the most likely word alignment for each, and then use only aligned words for count collection, we would expect to obtain better statistics for estimating the word translation probability distribution $t(e|f)$. In the next section, we will do one better: we will invoke the magic of the EM algorithm.

[4]The noisy channel approach employs the Bayes' rule $\text{argmax}_{\mathbf{e}} p(\mathbf{e}|\mathbf{f}) = p(\mathbf{e})\ p(\mathbf{f}|\mathbf{e})$ to integrate a language model $p(\mathbf{e})$, and hence reverses the direction of the translation model from $p(\mathbf{e}|\mathbf{f})$ to $p(\mathbf{f}|\mathbf{e})$.

1.3.3 Expectation Maximization

To learn word translation probabilities from a parallel corpus, we are suffering from the curse of incomplete data. Yes, we have a parallel corpus, where English sentences are matched up with their foreign translations. But while we have aligned sentences, we do not have aligned words.

If we had the true word alignment, it would be straightforward to collect counts and estimate word translation probabilities $t(e|f)$. On the other hand, if someone gave us the true word translation probabilities $t(e|f)$, then we could find for each sentence pair the most likely word alignment. But we have neither. So, what can we do?

The basic idea of the EM algorithm is the following: let us just pretend that we have a probability distribution $t(e|f)$. Then we can find the best word alignments. And with those word alignments, we can estimate a better model. Now we have a new model and can repeat the process.

The EM algorithm, in a nutshell, is the following process:

1. Initialize the model, typically with a uniform distribution

2. Apply the model to the data: compute probabilities for each possible word alignment

3. Learn the model from the data, based on collected counts from the word alignments and estimate a new word translation probability distribution

4. Go to steps 2 until convergence

In fact, in the section above, we already went through two iterations of the EM algorithm with slight simplifications. In the EM algorithm, we have to consider each possible alignment — not just the most likely alignment — and collect counts based on the conditional probability of the alignment given the sentence pair (we implicitly did this in the first iteration by collecting fractional counts).

Going over each possible alignment is quite a daunting task: since each English word is aligned to any of the foreign words, there is an exponential number of possible word alignments for a sentence pair. For IBM Model 1, there is a trick that allows us to do exact estimation in polynomial time, but in further refinements of the model (see section below), this is no longer possible. Instead, we have to sample the alignment space to find the most likely alignments, and restrict our count collection to this subset.

1.3.4 Alignment model

IBM Model 1 is a very simplistic model for word alignment or statistical machine translation, and even the original researchers at IBM only used it as a stepping stone to more sophisticated models. It does not work very well with rare words, since there are just too many choices. It also does not work if a foreign word occurs multiple times in the foreign sentence. Which one of them should an English word pick for alignment? They all have the same probability.

One way to extend the model is to include a component for alignment probabilities. IBM Model 2 introduces a model based on absolute word positions: $a(i|j, l_e, l_f)$. Based on the length of the English and foreign sentences l_e, l_f and the English word position j, we predict the foreign word position i.

Putting it all together, we have IBM Model 2:

$$p(\mathbf{e}, a|\mathbf{f}) = \frac{1}{Z} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})\ a(a(j)|j, l_e, l_f) \tag{1.10}$$

Instead of mapping absolute word positions, it is preferable to condition the alignment of a word on the alignment of its preceding word. After all, words typically move in phrases. Such a relative alignment model is used in IBM Model 4, and also in the HMM alignment model [Vogel et al., 1996].

One more extension: while we restrict each English word to align to a single foreign word, foreign words may align to any number of English words. To reign in, IBM Model 3 introduces the concept of **fertility**, and adds another conditional probability that predicts how many English words a foreign word generates.

1.3.5 Symmetrization

We have come to the point to admit to the most shameful secret in statistical machine translation. The IBM Models, although they are still commonly used for word alignment, are fundamentally flawed. The trick that makes EM training work so well is to enforce that each English word aligns to exactly one foreign word.[5]

Linguistically, this makes very little sense, and the restriction to 1-to-many alignments is also oddly asymmetrical. So, what can be done? Well, we run the EM training with the IBM Models in both directions (resulting in a

[5] Actually, we also allow an English word to align to the artificial NULL word, but we do not allow an English word to align to multiple foreign words.

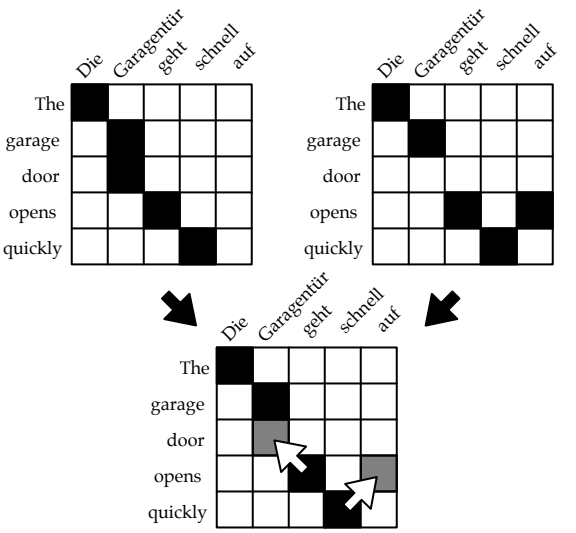

Figure 1.4. Overcoming the flaw in IBM Models, which assume one-to-many alignments (or many-to-one if run in reverse): A heuristic such as *grow-diag-final* starts with the alignment points from the intersection of two alignments obtained by running the model in both directions (black arrows). Then, it adds neighboring alignment points from the union (white arrows).

1-to-many and a many-to-1 alignment), and then force an agreement between the two resulting word alignments. We call this crude hack **symmetrization** [Och and Ney, 2003].

Once we obtained the two word alignments, we can take the union and the intersection between their alignment points. In a common method, we use all alignment points in the intersection and add some of the alignment points in the union. Typically points that neighbor already established alignment points are added. See Figure 1.4 for an illustration.

For instance, in the commonly used **grow-diag-final** method that ships with the open source Moses system, alignment points that directly and diagonally neighbor existing alignment points are added until convergence. Then, in a desperate final step, alignment points for words that are hitherto unaligned are added. The pseudo-code for the heuristic is given in Figure 1.5 on the facing page.

There are a number of refinements of this symmetrization process, for instance symmetrization can be done after each iteration of EM training [Matusov et al., 2004]. Also, machine learning methods have been developed

Input: sentence pair $(\mathbf{e}, \mathbf{f})$, with alignments $e2f$ and $f2e$
Output: word alignment **a**

grow-diag-final($e2f$,$f2e$):
neighboring = ((-1,0),(0,-1),(1,0),(0,1), (-1,-1),(-1,1),(1,-1),(1,1))
alignment = intersect($e2f$,$f2e$);
GROW-DIAG();
FINAL($e2f$);
FINAL($f2e$);

function GROW-DIAG():
while iterate until no new points added **do**
  for all english words $e \in \{e_1 ... e_n\}$ **do**
    for all foreign words $f \in \{f_1 ... f_m\}$ **do**
      if e aligned with f **then**
        for all neighboring points $e_{\text{new}}, f_{\text{new}}$ **do**
          if (e_{new} unaligned | f_{new} unaligned)
          & $(e_{\text{new}}, f_{\text{new}}) \in$ union$(e2f, f2e)$ **then**
            add alignment point $(e_{\text{new}}, f_{\text{new}})$
          end if
        end for
      end if
    end for
  end for
end while

function FINAL(a):
for all english words $e_{\text{new}} \in \{e_1 ... e_n\}$ **do**
  for all foreign words $f_{\text{new}} \in \{f_1 ... f_m\}$ **do**
    if (e_{new} unaligned | f_{new} unaligned) & $(e_{\text{new}}, f_{\text{new}}) \in$ union$(e2f, f2e)$ **then**
      add alignment point $(e_{\text{new}}, f_{\text{new}})$
    end if
  end for
end for

Figure 1.5. Pseudo-code for a symmetrization heuristic that settles on a set of alignment points between the intersection and the union of two IBM Model alignments.

to either iteratively add alignment points to the intersection [Ren et al., 2007], [Ma et al., 2008], or to delete alignment points from the union [Fossum et al., 2008].

1.3.6 Word alignment as machine learning problem

As with evaluation metrics, once the natural language community manages to properly define a problem, it does not take long before the hordes of machine learning researchers come in and attack it with all their favorite algorithms. This has also happened with word alignment, increasingly so in recent years.

For machine learning, word alignment is an interesting unsupervised learning problem. It would be futile to list all methods that have been recently applied. You have your usual suspects, as the perceptron algorithm [Moore, 2005], [Moore et al., 2006], maximum entropy models [Ittycheriah and Roukos, 2005], neural networks [Ayan et al., 2005], max-margin methods [Taskar et al., 2005], boosting [Wu and Wang, 2005], [Wu et al., 2006], support vector machines [Cherry and Lin, 2006], conditional random fields [Blunsom and Cohn, 2006], [Niehues and Vogel, 2008] or MIRA [Venkatapathy and Joshi, 2007].

One ingredient for a successful onslaught of machine learning was the establishment of test sets, in this case gold standard word alignments that were created by human annotators. There are several such sets for a number of language pairs, usually available through the LDC.[6]

There is some debate over how to best evaluate alignment quality. One early metric, alignment error rate (AER) has come under heavy criticism [Fraser and Marcu, 2007]. Since word alignment is mostly done in the context of statistical machine translation, the ultimate evaluation metric is the machine translation quality that can be obtained using the word alignment. Of course, this is a very costly metric to compute.

1.4 Phrase-based models

Currently, the dominant approach in statistical machine translation is a model based on the mapping of short text chunks (typically only 1-3 words long), which are somewhat misleadingly called phrases albeit they are not necessarily *linguistic* phrases, i.e. constituents in a syntactic analysis.

Compared to word based models, phrase-based models overcome the fundamental flaw of insisting on the lexical mapping of a 1-to-1 correspondence

[6] http://www.ldc.upenn.edu/

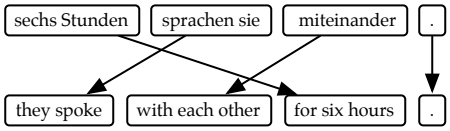

Figure 1.6. Phrase-based machine translation. The input is segmented into phrases (not necessarily linguistically motivated), translated one-to-one into phrases in English and possibly reordered.

of words. Of course, word-based models do confront reality and introduce components such as fertility or NULL-word generation. But these complications make training and decoding algorithms much more cumbersome. Phrase-based models also have the advantage that with more training data longer and longer phrases can be learned. In the limit, a sentence may be translated by looking it up in its entirety in the training corpus.

1.4.1 Model

The phrase-based model has the advantage that it is quite easy, which allows for straightforward training methods and efficient decoding algorithms. See Figure 1.6 for an illustration. The input sentence is segmented into phrases and each phrase is mapped 1-to-1 into an English phrase. Phrases may be reordered.

Let us now define the phrase-based statistical machine translation model mathematically. First, we apply the Bayes' rule, so we can integrate a language model p_{LM} by inverting the translation direction. The best English translation $\mathbf{e}_{\text{best}}$ for a foreign input sentence $\mathbf{f}$ is defined as

$$\begin{aligned} \mathbf{e}_{\text{best}} &= \text{argmax}_{\mathbf{e}}\ p(\mathbf{e}|\mathbf{f}) \\ &= \text{argmax}_{\mathbf{e}}\ \frac{p(\mathbf{f}|\mathbf{e})\ p_{\text{LM}}(\mathbf{e})}{p(\mathbf{f})} \\ &= \text{argmax}_{\mathbf{e}}\ p(\mathbf{f}|\mathbf{e})\ p_{\text{LM}}(\mathbf{e}) \end{aligned} \tag{1.11}$$

Note that we can ignore $p(\mathbf{f})$ since it is constant for all possible translations $\mathbf{e}$. We decompose $p(\mathbf{f}|\mathbf{e})$ further into

$$p(\mathbf{f}|\mathbf{e}) = \prod_{i=1}^{I} \phi(\bar{f}_i|\bar{e}_i)\ d(start_i - end_{i-1} - 1) \tag{1.12}$$

The foreign sentence $\mathbf{f}$ is broken up into I phrases $\bar{f}_i$. Each foreign phrase $\bar{f}_i$ is translated into an English phrase $\bar{e}_i$. Since we mathematically

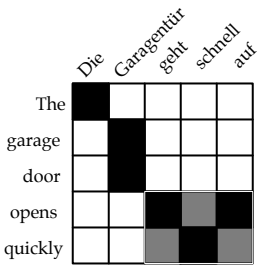

Figure 1.7. Phrase extraction: given the word alignment, the phrase pair (*opens quickly*, *geht schnell auf)* is extracted.

inverted the translation direction, the phrase translation probability $\phi(\bar{f}_i|\bar{e}_i)$ is modelled as a translation from English to foreign.

Reordering is handled by a **distance-based reordering model**. We consider reordering relative to the previous phrase. We define $start_i$ as the position of the first word of the foreign input phrase that translates to the ith English phrase, and end_i as the position of the last word of that foreign phrase. Typically this model is not estimated from the data, but rather a fixed cost relative to the movement distance is applied: $d(x) = \frac{1}{Z}\alpha^{|x|}$.

There are other components that we may want to add. Typically, a word penalty $\omega^{|e_i|}$ is factored in that adds a factor ω for each produced word. This allows us to adjust the model to produce shorter or longer output.

1.4.2 Training

The main knowledge source in phrase-based models is a massive phrase translation table. It contains input phrases, their possible translations and corresponding probability scores.

The table is learned from a word-aligned parallel corpus. Given a sentence pair and a word alignment, we extract all phrase pairs that are consistent with the word alignment. By consistent we mean that all the words in the phrases align to each other and not to words outside the phrase pair.

See Figure 1.7 for an example. Given the word alignment points between *(opens, geht)*, *(opens, auf)*, and *(quickly, schnell)*, we can extract the phrase pair (*opens quickly*, *geht schnell auf)*.

There are some subtle decisions in phrase extraction, for instance a maximum phrase length (typically 5–7), if phrases may have unaligned words at the boundaries (typically yes, but sometimes limited), or if partial counts are assigned, if multiple target phrases are found for a source phrase (done

either way).

When extracting all phrase pairs, count collection leads to a straightforward definition of conditional phrase translation probability based on relative frequency:

$$\widehat{\phi}(\bar{f}|\bar{e}) = \frac{\text{count}(\bar{e}, \bar{f})}{\sum_{\bar{f}'} \text{count}(\bar{e}, \bar{f}')} \tag{1.13}$$

The conditional phrase translation probabilities are often estimated from very sparse counts. In the extreme case, for an English phrase $\bar{e}$ that only occurs once, its lone foreign correspondent $\bar{f}$ receives a phrase translation probability of $\widehat{\phi}(\bar{f}|\bar{e}) = 1$.

There are several ways to remedy this. Commonly, additional scoring functions based on the lexical translation probabilities are added, for instance IBM Model 1. Discounting the raw counts using Good Turing smoothing have also been shown to be effective [Foster et al., 2006].

We will refine the model in Section 1.4.5 on page 23 below, where we reformulate it as a log-linear model that allows for the easier integration of additional scoring functions, but let us first turn to the practical issue of actually producing translations for a new, previously unseen input sentence.

1.4.3 Decoding

Let us say, we would like to translate the German sentence

Sechs	*Stunden*	*sprachen*	*sie*	*miteinander*	.
six	*hours*	*spoke*	*they*	*with each other*	.

An English sentence typically starts with the subject, so when creating a translation into English, we would pick out the subject *sie* and start the translation with *They*. Then we would pick out the verb *sprachen* and continue to the translation with *spoke*. By building the translation from left to right, we reach the English translation:

They spoke with each other for six hours.

With our decoding algorithm, we want the machine also to translate from left to right. However, this is not so simple. There are many options in our phrase translation table to choose from. See Figure 1.8 on the following page for an excerpt of the options from a real example (using a phrase translation table aquired from the Europarl corpus). Only when we have the full sentence translation, then we can compute its full translation probability.

Sechs	Stunden	sprachen	sie	miteinander	.
six	hours	it would be		with each other .	
six ,	hours ,	it would		to each other .	
6	hours of	they spoke		together	.
for six	few hours	spoke	they	with	
	time	talked	she	each other	

Figure 1.8. Translation options for a short German sentence

At the beginning, the algorithm may pick any of the translation options displayed in the figure. In the second step, we are not able to translate the same input word or phrase again, but we are still left with almost as many options. A naïve algorithm that tries all possible combinations of translation options to translate the sentence has a run time that is exponential with respect to sentence length. In fact, machine translation decoding has been proven to be NP-complete [Knight, 1999].

In the commonly used beam search stack decoding algorithm, we explore the space of possible translations by keeping the promising partial translations and extending them with new translation options until we have covered the entire input sentence.

The partial translations (called **hypotheses**) are organized in **stacks**, based on the number of input words they cover. So, for instance, stack one contains all hypotheses that have already translated one input word. We only keep a limited number of hypotheses in a stack, so we have to prune out some that do not look promising.

We proceed through the expansions of all hypotheses in a stack, which generates new hyoptheses that are placed in stacks farther down the road. Then, we move on to the next stack. The pseudo code for the algorithm is displayed in Figure 1.9 on the next page. For a graphical illustration, please refer to Figure 1.10 on the facing page.

The term **beam search** implies that we search through the most promising part of the search space, using a "beam of light" that illuminates a number of alternatives but that is not bright enough to explore all possible paths.

We mentioned that we have to sort out the less promising hypotheses in each stack. Note that when generating a partial translation, we can already compute all the probability costs implied by Equation 1.13 on the previous page that have been incurred by the translation options so far. We can then sort the hypotheses according to the costs-so-far and drop the worst ones.

Input: Foreign sentence $\mathbf{f} = f_1, ...f_{l_f}$
Output: English translation **e**

```
place empty hypothesis into stack 0
for all stacks 0...n − 1 do
    for all hypotheses in stack do
        for all translation options do
            if applicable then
                create new hypothesis
                place in stack
                recombine with existing hypothesis if possible
                prune stack if too big
            end if
        end for
    end for
end for
```

Figure 1.9. Pseudo-code for the stack decoding heuristic.

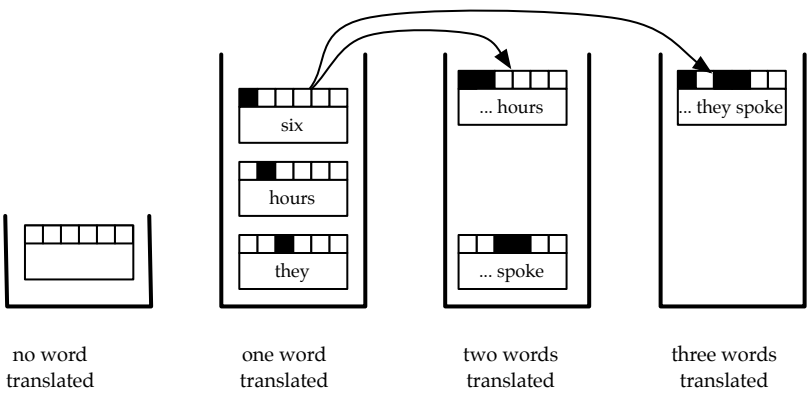

Figure 1.10. Illustration of the stack decoding heuristic.

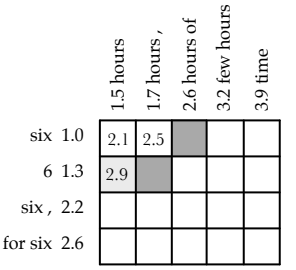

Figure 1.11. Cube pruning: Sorting hypotheses (y-axis) and options (x-axis) allows for the expansion of only the most promising hypotheses.

Especially since hypotheses may cover different words, pruning based on costs-so-far is unfair to the translations that already tackled the harder part of the sentence. So, in addition to the cost-so-far, we include a **future cost estimate**.

There is one more, very important twist: **recombination**. There may be two different decoding paths that lead to pretty much the same state in the search. For instance, we may start translation with the 1-word phrase mapping *sie* → *they* and then continue with *sprachen* → *spoke*. But we may have simple used the 2-word phrase mapping *sprachen sie* → *they spoke*. One of the two resulting hypotheses will have a higher cost-so-far (depending on the translation costs of the phrase translations), so we can safely drop the worse one.

Note that hypotheses do not have to match exactly for recombination, they only have to be undistinguishable in terms of their continuation. While they have to match in the foreign word coverage (this affects future search), they may differ in the output words that they have produced so far if these are outside the window of the n-gram language model.

1.4.4 Cube Pruning

A popular new variation of the decoding heuristic is called **cube pruning**, although in phrase-based decoding it has nothing to do with cubes or pruning. A better name would be **sorted expansion**. Since most of the hypothesis that are generated are discarded, cube pruning focuses on expanding hypotheses that are most promising. To this end, hypotheses are translation options are sorted, and only the most probable hypotheses are combined with the most probable translation options.

See Figure 1.11 on the facing page for an illustration. Let us say that we want to expand hypotheses that covered the first word already with translation options that cover the second word. In the example, there are four such hypotheses, and five such translation options - in practice the numbers are easily a magnitude bigger. The original beam search algorithm generates all four×five expansions — we now want to focus on just a subset of these.

The most promising new hypotheses is the combination of the best old hypothesis with the best translation option. So, we start in the top left corner.

We could proceed by creating the top n hypotheses based on the old hypothesis cost and the estimated translation option cost. However, the cost of the new hypothesis is not simply the combination of the cost of the hypothesis and the cost of the expansion. Only once we put them together, we can compute the true language model cost, and we know the true hypothesis cost.

So, instead, we proceed in a way that takes into account that some hypothesis and some translation options turn out to be more promising that their costs suggest. We always expand the hypotheses are neighbor the best hypothesis that we have expanded so far and that still has unexpanded neighbors.

In the example, the most promising hypothesis in the top left corner has a true cost of 2.1. We explore its neighbors, which have costs of 2.5 and 2.9. Next we, explore the neighbors of the hypothesis with cost 2.5, and so on.

1.4.5 Log-linear models and parameter tuning

We have already mentioned a number of components that may improve our translation model: lexical translation probabilities or a word penalty. It would be awkward to derive a model that includes such components mathematically from the sentence translation probability $p(\mathbf{e}|\mathbf{f})$ with appeals to interpolation, independence assumptions and back-off, so may just throw in the towel, and clearly state that our model is a combination of feature functions h_i that are weighted (λ_i) according to their importance:

$$\begin{aligned} p(\mathbf{e}|\mathbf{f}) &= \prod_i h_i(\mathbf{e},\mathbf{f})^{\lambda_i} \\ \log p(\mathbf{e}|\mathbf{f}) &= \sum_i \lambda_i \log h_i(\mathbf{e},\mathbf{f}) \end{aligned} \tag{1.14}$$

The feature functions are the components of the phrase-based model that

we introduced in Section 1.4.1 on page 17, for instance the language model $h_{\text{LM}}(\mathbf{e}, \mathbf{f}) = p_{\text{LM}}$, or the phrase translation model $h_\phi(\mathbf{e}, \mathbf{f}) = \sum_i \phi(\bar{f}_i|\bar{e}_i)$.

Now that we introduced weights λ_i for the feature functions h_i, how do we set them? Each of the feature functions has something to say about the English sentence **e** being a good translation of the **f**. We would like to weight these functions so that we optimize overall translation quality.

Here is where we close the circle to the discussion of automatic evaluation metrics: for a given **tuning set** of foreign input sentences and their reference translations, we can — under any given weight setting $\{\lambda_i\}$ — translate the set with our model and decoder, and compute the resulting automatic BLEU score. We then change the weight setting, decode again, and see if we have an improvement. What we have here is a well defined multi-dimensional optimization problem, which is called **parameter tuning** or **minimum error rate training** (MERT).

Since decoding is very expensive, we use a short-cut of generating n-best translations for each input sentence and carry out optimization on these n-best lists first. A common method [Och, 2003] optimizes one parameter at a time. It is possible to find the optimal value for one parameter when leaving all others fixed. However, we are limiting ourselves to a grid search that may get stuck in local optima. So, random restarts are needed. Also, we re-run the decoder to avoid optima on the n-best lists that are unrepresentative of the full search space.

1.4.6 Coping with model size

It is not entirely intuitive, but the phrase translation tables we build for phrase-based models are much larger than the parallel corpus itself. Consider that a sentence with n contains $O(n^2)$ phrases.

Typically, training corpora have millions of sentence pairs, and resulting phrase translation tables are often measured in Gigabytes. Moore's Law has gone a long way to make the practical use of phrase based models feasable, but even today we are not able to store large models in memory. Efforts to run translation systems in hand-held devices excerbates the situation.

A number of solutions have been suggested, ranging from efficient storage of the translation table to filtering and pruning. Let us take a look.

We already mentioned that a table is larger than the corpus we extract it from. A compelling solution is to not store the table at all, but only the corpus. Of course, we need to quickly look up source phrases (and their translations) that match a given input sentence. Here, the use of suffix arrays has been proposed [Callison-Burch et al., 2005].

A suffix array is a data structure that contains a sorted list of all suffixes of the corpus. Think of suffixes as very long phrases starting anywhere in the corpus and going to the end. The number of suffixes is identical with the number of words in the corpus, and hence the sorted index is of the same size as well. When we look up a suffix of the input sentences, we can find arbitrarily large matches using the index. We then we refer to the (also stored) word alignment and target side of the corpus to extract phrases on the fly.

However, if even the corpus size is too large, then we need to be even more constraint in what we store in memory. Note that, for the translation of a single sentence, only a tiny fraction of the phrase translation table is used. Instead of loading all of it into memory, we can filter it down to the fraction that is needed. Filtering is common in experimental work, where the same test set of typically one or two thousand sentences is used.

If we want to deploy a machine translation system in an online setting, we do not have the time to filter through Gigabytes of data. Unless, we organize the table on-disk in an efficient data structure that lends itself to quick lookups of phrases, such as a prefix tree [Zens and Ney, 2007].

Finally, we can take a hard look at the translation table, realize that much of it is junk: either long phrase pairs that are very unlikely to ever be useful, or low-probability phrases such as thousands of translations for the period (there are even more for the comma). So, why do not just clean up? Phrase pairs may be discarded based on significance tests on their more-than-random occurrence [Johnson et al., 2007], log likelihood ratios [Wu and Wang, 2007]. Such considerations may also be taking into account in second pass phrase extraction stage that does not extract bad phrase pairs [Zettlemoyer and Moore, 2007].

We may only need to extract the shortest phrase pairs that explain each training sentence pair [Quirk and Menezes, 2006]. This is also the basis of the n-gram translation model [Mariño et al., 2006], [Costa-jussà et al., 2007], a variant of the phrase-based model. Or, we may prune the translation table based on how often a phrase pair was considered during decoding and how often it was used in the best translation [Eck et al., 2007a], [Eck et al., 2007b]. Finally, [Kutsumi et al., 2005] uses a support vector machine for cleaning phrase tables.

1.5 Tree-based models

Any reader with some background in linguistics will view our models as hopelessly naïve. One of the fundamental properties of language is recursion.

A sentence is made up of clauses, which are made up of verbs, noun phrases and alike. Noun phrases may include relative clauses that again are made up of verbs etc. Virtually all modern theories of grammar look at a sentence and do not see a string of words, but a hierarchical tree structure.

None of this is a revelation to researchers in statistical machine translation. The use of syntactic trees — either using syntactic parsers or automatically learning tree structures from the data — within the paradigm of statistical machine translation has been an ongoing focus of attention since the mid-1990s. However, until very recently, such approaches have not been successful in head-to-head comparisons with simpler phrase-based models.

One reason is that operations on tree structures are more complex and thus require computationally more expensive learning methods and also make the search during decoding much harder. This led to simplifications such as requiring some form of isomorphism between source and target syntax trees (for instance, only allowing reordering of children nodes, but no major restructuring) that turned out to be too harsh a restriction.

Another problem for syntax-based approaches is that talking about syntax in all its glory is fine, but at the end of day we have to use available syntactic parsing tools that may be just not good enough.

Current tree-based approaches build on the success of phrase-based models, and can be seen as extensions of that approach.

1.5.1 Hierarchical phrase-based modes

A limitation of phrase-based models, as we defined them, is that they do not allow for discontinuous phrases with gaps. For instance, we may want to map between English and French:

$$\textit{does not } X \rightarrow \textit{ne } X \textit{ pas}$$

Such a mapping may be expressed, however, by a synchronous context-free grammar, which distinguished between terminal symbols (words) and non-terminals (X). A grammar rule may also have multiple non-terminals:

$$X_1 \textit{ of } X_2 \rightarrow X_2 \; X_1$$

Coming from phrase-based models, we may view such rules as phrase mappings from which sub-phrase pairs have been subtracted. Allowing for such rules, we obtain better explanation for certain reordering phenomena, role of functions words such as *of* and discontinuous phrases.

The extraction of such so-called **hierarchical phrase pairs** [Chiang, 2005] from a word aligned parallel corpus is straight forward. In addition to

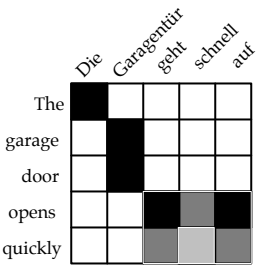

Figure 1.12. Learning hierarchical phrase translation rules: Starting with the phrase pair *(geht schnell auf, opens quickly)*, we extract the sub-phrase pair *(schnell, quickly)* and arrive at the translation rule *(geht* X *auf, opens* X*)*.

the fully lexicalized phrase pairs, we have to take a look at each phrase pair, see if we can subtract sub-phrase pairs and replace them by non-terminals. Then, we proceed to also add these hierarchical phrase pairs to our translation table. See Figure 1.12 for an example.

The process of extracting sub-phrase pairs has the potential to blow up the number of phrase pairs, we have to introduce some reasonable constraints, for instance: rules must include at least one word, rules must span at most a maximum number of words, etc.

Adding hierarchical phrase pairs seems to be an obvious win, except that it breaks the decoding algorithm we presented in Section 1.4.3 on page 19, which required that we construct the translation left to right. How can we build a sentence left to right, when we have to add phrases such as *ne X pas*?

Coming from syntactic grammars, there is a straight forward answer: this is a parsing problem, and we have to apply a parsing algorithm, such as chart parsing.

1.5.2 Chart decoding

Instead of decoding from left to right, we decode bottom up. First we find translations for all single words, then for all spans of two words, then for all spans of three words, and so on until we covered the entire sentence. See Figure 1.13 on the following page for an illustration.

For instance, when translating the sentence

Je ne parle pas anglais.

into English, we may first apply a number of traditional phrase translation rules that give us the chart entries:

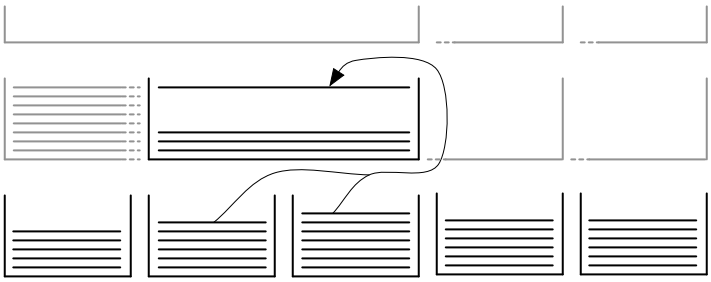

Figure 1.13. Decoding with tree-based model: One stack for each input word (bottom row), and higher level stacks for each contiguous span.

I		*speak*		*English*
je	*ne*	*parle*	*pas*	*anglais*

We can then apply hierarchical phrase rule (where X matches *speak*):

$$ne\ X\ pas \rightarrow do\ not\ X$$

This adds the chart entry *do not speak*:

	do not speak			
I		*speak*		*English*
je	*ne*	*parle*	*pas*	*anglais*

Finally, we can apply the glue rule:

$$X_1\ X_2 \rightarrow X_1\ X_2$$

Using it twice gives us the complete output:

I do not speak English				
I do not speak				
	do not speak			
I		*speak*		*English*
je	*ne*	*parle*	*pas*	*anglais*

A rough sketch of the chart decoding algorithm is given in Figure 1.14 on the next page. In practice, a number of refinements are needed to avoid the loop over all possible sequences (starting in line 4) and all rules (starting in line 5), which is computationally too expensive. When entering new chart entries for a span, an efficient search through the underlying chart entries and available rules has to be carried out, for instance by using Early parsing.

Input: Foreign sentence $\mathbf{f} = f_1, ...f_{l_f}$
Output: English translation **e**

```
for span length l = 1 to l_f do
  for start=0 .. l_f-l do // beginning of span
    end = start+l
    for all sequences s of entries and words in span [start,end] do
      for all rules r do
        if rule r applies to chart sequence s then
          create new chart entry c
          add chart entry c to chart
        end if
      end for
    end for
  end for
end for
return English translation e from best chart entry in span [0,l_f]
```

Figure 1.14. Sketch of the core chart decoding algorithm

1.5.3 Syntactic models

Having made the leap to hierarchical phrase models, it is not much further to a syntax-based model with real constituents labels such as VP and NP. In addition to a word aligned parallel corpus, we now also need syntactic annotation on source or target side, or both. See Figure 1.15 on the following page for an example.

Syntax on the source side acts as a restriction on which rules may apply. Syntax on the target side requires the syntactic parsing of the output sentence into a tree, thus enforcing syntactic well-formedness in addition to the fluency enforced by the n-gram language model.

To give one example, our French verb negation rule may look like this:

$$\text{VP: } ne \text{ V } pas \rightarrow \text{VP: } do \text{ } not \text{ V}$$

There are some caveats to keep in mind when adding syntactic labels. Phrases in phrase-based models do not have to match constituents in a syntax tree. However, since each chart entry requires a constituent label, we have two choices: (a) require that the target side phrase is a single constituent, or (b) create artificial constituent labels.

Let us give an example. In a phrase models, we may have the rule:

$$der \text{ } große \rightarrow the \text{ } big$$

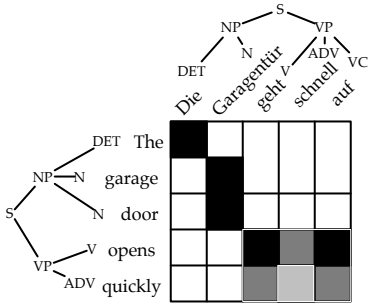

Figure 1.15. Extracting syntactic translation rules: As in the hierarchical phrase pair example from Figure 1.12 on page 27, the phrase pair *(geht* X *auf, opens* X*)* is acquired. The syntactic annotation informs the identity of the non-terminals, thus giving us the rule VP: *geht* ADV *auf* → VP: *opens* ADV.

When adding syntactic labels on the English target side, choice (a) requires us to extend the rule to the entire noun phrase that includes *the big*:

$$\textit{der große}\ \text{X} \rightarrow \text{NP}\textit{: the big}\ \text{N}$$

Choice (b) requires us to make a new label, for instance:

$$\textit{der große} \rightarrow \text{DT+J}\textit{: the big}$$

Both choices have draw-backs: choice (a) throws out many phrase pairs as potential rules, thus limiting the knowledge we extract from the parallel corpus. Choice (b) leads to an explosion of non-terminal labels, thus making decoding harder.

A third choice that has been used in syntax-based models is the binarization of the syntax tree to make the restriction of limiting phrases to constituents less harsh.

1.6 Linguistic challenges

So far, we have only paid scant attention to the nature of the problem of translation. Most readers, especially if they have learned a second language, will have an intuitive understanding of what makes translation a hard problem: words in the source language have different meanings and thus different translations, the word order between languages differs, and the relationship between words in a sentence are encoded in different ways — be it morphological markup, function words and order.

All these problems have to be overcome in statistical machine translation systems. While we pretend that the methods that we present here are language-independent, they do in fact work better if the languages between we translate have mostly the same word order, similar concepts and metaphors, and there is little morphological complexity on the target side. To give an example, they work very well for French–English, but perform worse for Chinese–Turkish.

1.6.1 Lexical choice

A popular problem in computational linguistics is word sense disambiguation. Words such as *interest* or *bank* have multiple meanings. This shows up in translation as the problem of lexical choice, i.e., we have to chose as translation for *bank* into German either *Bank* or *Ufer*, thus selecting either the money or river sense of the word.

Research in word sense disambiguation has shown that local context (the neighboring words or part-of-speech tags), content words in a larger window, the syntactic role of the word, and syntactic related words are good indicators of word sense.

In fact, by using an n-gram language model, we already capture effectively local context information that is very useful to make the right lexical choice. The prior probability is also very helpful: the money sense of *bank* is just much more frequent than its river sense. Consequently, statistical machine translation systems handle lexical choice rather well, certainly better than the traditional rule-based systems.

Nevertheless, in recent years, some researchers have targeted word sense disambiguation in statistical machine translation and have shown gains by incorporating the type of additional features we alluded to above. It is very straightforward to convert the traditional conditional probability distribution — be it at the word or phrase level — into a more sophisticated classifier.

A common choice is to use maximum entropy to incorporate arbitrary features from the source sentence. It is harder to incorporate target side features, since these often split states in our beam search decoding algorithm. If we, say, condition the translation of a word on the first word in our target sentence, we cannot recombine hypotheses that have different first words.

1.6.2 Morphology

The models we have presented so far operate on surface forms of words. For instance, they make no connection between the singular *house* and plural

houses. Since most of the work in statistical machine translation has focused on English as a target language and English has relatively low morphological complexity, this has not been regarded as a high priority problem. Yes, we lose some generalization ability by treating *house* and *houses* as two completely different words, but we also keep our model simple and may be able to make sometimes better distinctions in the translation of singular and plural forms.

However, when translating into morphologically rich languages such as Turkish, Hungarian, Czech, or German, treating morphology becomes much more important. The first concern is that rich morphology leads to much larger vocabulary sizes and hence sparse data problems in model estimation.

Secondly, when translating into morphologically rich languages, it is often not clear from the local context which morphological variant to chose. For instance, when translating *the man* into German, we have as choice *der Mann*, *des Mannes*, *dem Manne*, or *den Mann*. Which one is correct depends on the relationship of the noun phrase to its syntactic head, e.g., is it the subject or object?

Factored translation models [Koehn and Hoang, 2007] propose to represent words not as simple tokens but as vectors of factors such as the lemma, part-of-speech tag, gender, count, and so on. Including such additional annotation into our models has two benefits. First, it allows for generalization such as the translation between lemmas instead of surface forms. Secondly, it enriches the model which can be exploited for, say, reordering based on part-of-speech tags or grammatical coherence checks based on morphological tags.

Adding factored representations to phrase models has been used to enrich the input for better morphological choice in the target, increase grammatical coherence in the output, or improve translation of rare morphological variants. The work has also shown that there is a risk involved in breaking up phrase translation into separate mapping steps by assuming independence between them. If we know how to translate morphologically rich phrases because we have seen them very often, there is only harm in decomposing their translation into more fine-grained steps — just as the translation of large phrases (if possible) is better than word-for-word translation.

1.6.3 Word order

Sentences are made up of one or more clauses, and each clause describes an action that centers on a verb and its arguments and adjuncts. To define which of the entities mentioned in the sentence is the subject and which are

the objects and what they roles are, language such as English use word order.

English is an SVO language, meaning that a clause typically starts with a subject (*English*), followed by a verb (*is*) and any number of objects (*an SVO language*). Other languages may have different canonical word order, for instance VSO, SOV, etc. This presents a straightforward problem for translation: the words need to be rearranged when mapped into the target language.

The insight that reordering is mostly driven by syntactic differences is one fundamental motivation for the tree-based models that we discussed in Section 1.5 on page 25. If we obtain syntactic trees of the input or construct parse trees of the output during translation, then movements that look arbitrary on the surface level (e.g., a word moves nine positions to the left), may be a simple child node reordering in a syntactic tree.

Since the use of tree-based models is generally rather complex, simpler approaches have been proposed to incorporate syntactic trees into statistical machine translation models. One idea is to **pre-reorder** the input before the actual translation process. The goal is to re-arrange the input in a way that it still consists of the same input words, but now in the expected word order of the output. This may be done via hand-written rules (since we are mostly worried about well-understood long-distance movements), or rules learned automatically from a word-aligned source-annotated parallel corpus, maybe even just using part-of-speech tags. We may settle on a unique input sequence or represent potential choices in a reordering lattice. The expectation is that such a rearranged input is easier to translate with traditional phrase-based models, maybe even without allowing any reordering at all afterwards.

Another class of languages, free word order languages, cannot be easily classified as SVO or VSO. Recall that the purpose of a fixed word order is to define the relationship between the different constituents in the sentence, such as the relationships of noun phrases to the verb. Some languages use different means to define this relationship: markers or noun cases. Markers are used for instance in Japanese, but also English speakers should be familiar with the concept: prepositions play pretty much the same role (*from the house* vs. *to the house*). Noun cases change the surface form of words, for instance *der Mann* is a subject, but *dem Manne* is an object.

The mapping between languages that use such different means of defining syntactic relationships has not yet been sufficiently explored in statistical machine translation — partly because the strong focus on English as the output language where fixed word order is well handled by n-gram language models.

1.7 Tools and data Resources

Although building a machine translation system is a complex task, it is much facilitated by the tools and data resources that have been made available, often in form of open source software. Look out for any recent developments, but let us quickly review some of the most commonly used resources.

1.7.1 Basic tools

The training pipeline for statistical machine translation is fairly straight forward to implement, save for two non-trivial steps: sentence alignment and word alignment.

Translated text found out in the wild (think about a book and its translation, or a multi-lingual web site) rarely comes in the sentence-aligned format that the methods described in this chapter require. Hence, the first step is to align individual sentences that are translations of each other.

The simplest methods rely on length of sentences as a measure of similarity, more complex methods utilize also translation dictionaries. A widely used for sentence alignment is **Hunalign**[7], which uses both of these knowledge sources to determine the best alignments, and also offers filtering functions for potential mismatches.

We discussed the problem of word alignment at length in Section 1.3 on page 9. The **GIZA++** toolkit[8] is a open source implementation of the popular IBM Models that we presented above. It is widely used. More recently, the problem word alignment has regained attention in the research community. One outcome of this is the **Berkeley word aligner**,[9] which integrates the idea of symmetrizing word alignments (recall Section 1.3.5 on page 13) more closely into the alignment method.

The use of language models is essential for machine translation. In almost all cases, machine translation systems integrate existing language model tools and libraries, instead of re-inventing the wheel. Most popular is the open source **SRILM** toolkit,[10] which has been developed for over a decade. A recent addition is the **IRSTLM** toolkit,[11] which targets compact representation and scalable training tools for very large language models (using billions of words). Worth mentioning is also the **randLM** toolkit[12] which

[7] http://mokk.bme.hu/resources/hunalign/
[8] http://www.fjoch.com/GIZA++.html
[9] http://nlp.cs.berkeley.edu/Main.html#WordAligner
[10] http://www.speech.sri.com/projects/srilm/
[11] http://hlt.fbk.eu/en/irstlm/
[12] http://sourceforge.net/projects/randlm/

uses a lossy data structure to even more space-efficiently encode such large language models.

1.7.2 Machine translation systems

Entire machine translation systems — including the training process and the decoder — have become available with open source licenses.

The mostly used toolkit is **Moses**,[13] which implements most of the methods describes in this chapter. It draws heavily on existing tools for word alignment and language modeling mentioned above. More recently, the **Joshua**[14] decoder has been developed with a focus on hierarchical and syntax-based models.

We have not discussed rule-based approaches to machine translation in this chapter, but many commercial systems in use today are still based on translation rules written by hand. Typically, such systems allow the integration of more detailed knowledge into translation decision, but suffer from the lack of a language model and other probabilistically weighted decision processes. Nevertheless, there is still active work in this area. The open source **Apertium** project[15] aims at the construction of rule-based machine translation system for many language pairs.

1.7.3 Parallel corpora

Finally, or better firstly, you will need to have translated texts as training data for statistical machine translation systems — the more, the better. The more closely tailored to your domain of interest, the better.

Practically all parallel corpora used in machine translation systems are *found* corpora, i.e. they have been build for other purposes and co-opted by machine translation research. The main source of such corpora are governments (for instance Canada for French-English) and international institutions (UN, European Union). Most translations are currently produced in the commercial sector (product documentation, marketing material), but they are usually closely guarded by their content owners. A promising new direction is the exploitation of collaborative efforts on the internet to create translations — the buzz words are wiki translation and crowd sourcing.

Here a short list of commonly used corpora:

[13] `http://www.statmt.org/moses/`
[14] `http://sourceforge.net/projects/joshua/`
[15] `http://www.apertium.org/`

- The **Canadian Hansards**[16] consists of the proceedings of the Canadian parliament, which are translated between French and English.
- The **Europarl corpus**[17] consists of the translated proceedings of the European parliament. It offers about 40 million words in 11 languages each.
- The **Acquis corpus**[18] consists of the legal documents that member countries of the European Union have to sign up. The corpus covers 22 languages with up to 40 million words per language.
- The **OPUS project**[19] collects parallel corpora from a wide variety sources, including open source documentation and movie subtitles.
- The **LDC**[20] is the main source of data for the field of computational linguistics. The organization makes also parallel corpora available, especially for Arabic–English and Chinese–English, which have been the target of recent US sponsored research programs.

1.8 Future directions

The field of statistical machine translation is, despite of its 20 years of history, very much in motion. The strong focus on evaluation campaigns and hence emphasis of performance over fanciful ideas leads to rapid adoptions of new methods that have shown to be successful. Current research focuses on a number of issues that we briefly touch upon here.

Statistical machine translation models have a lot of numbers in them, and the estimations of these parameter values is a core problem. While the reliance on probability distributions mirrored in the training data has brought us a long way, there is intense interest in applying more advanced machine learning methods. Current systems rely on a mix of generative models (e.g., the phrase translation probabilities) and discriminate training (parameter tuning, see Section 1.4.5 on page 23).

Research on syntactic models is in full swing, and there are many open questions about how representation, e.g., phrase structure grammars vs. de-

[16] a part of the corpus is available at `http://www.isi.edu/natural-language/download/hansard/`, more is available through the LDC

[17] `http://www.statmt.org/europarl/`

[18] `http://wt.jrc.it/lt/Acquis/`

[19] `http://urd.let.rug.nl/tiedeman/OPUS/`

[20] `http://www.ldc.upenn.edu/`

pendency grammars, specific grammar formalisms, efficient decoding algorithms and so on.

The reliance on parallel data to train statistical models places a lot of importance on a often very scarce resource. How can be also utilize comparable or purely monolingual data? How do we deal with small in-domain data vs. large out-of-domain data? Can we exploit user interactions with machine translation systems as additional training data to improve our systems?

Finally, since machine translation closely interacts with other information processing applications, there is keen interest in integrating statistical machine translation into such applications.

Two applications have been explored in recent work. First, speech translation aims at the integration of speech recognition, machine translation, and speech synthesis. Secondly, since high quality translation still requires a human in the loop, recent computer aided translation tools exploit methods in statistical translations.

1.9 Summary

The application of data-driven methods to machine translation has turned the field into a hotbed of activity.

Machine translation is a seemingly stright-forward task: translating text written in one language into text in another language, but with the same meaning. But exactly how to measure, if a translation of a sentence is correct is still a open question, as we discussed in Section 1.2 on evaluation.

An important step in learning translation models from a parallel corpus is word alignment (Section 1.3). A word aligned parallel corpus allows the estimation of phrase-based (Section 1.4) and tree-based models (Section 1.5), the currently most commonly followed approaches.

Machine translation has made great progress, for instance translating news stories from French to English with today's technogoly produces very readable and accurate output. But many challenges remain, especially for language pairs with divergent word order and rich morphology (Section 1.6).

Research in the area is facilitated by a large array of open source tools and resources (Section 1.7), and many future directions remain to be explored (Section 1.8).

BIBLIOGRAPHY

[Ayan et al., 2005] Ayan, N. F., Dorr, B. J., and Monz, C. (2005). NeurAlign: Combining word alignments using neural networks. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 65–72, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

[Banerjee and Lavie, 2005] Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

[Blunsom and Cohn, 2006] Blunsom, P. and Cohn, T. (2006). Discriminative word alignment with conditional random fields. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 65–72, Sydney, Australia. Association for Computational Linguistics.

[Brown et al., 1988] Brown, P. F., Cocke, J., Della-Pietra, S. A., Della-Pietra, V. J., Jelinek, F., Mercer, R. L., and Rossin, P. (1988). A statistical approach to language translation. In *Proceedings of the International Conference on Computational Linguistics (COLING)*.

[Callison-Burch et al., 2005] Callison-Burch, C., Bannard, C., and Schroeder, J. (2005). Scaling phrase-based statistical machine translation to larger corpora and longer phrases. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 255–262, Ann Arbor, Michigan. Association for Computational Linguistics.

[Callison-Burch et al., 2008] Callison-Burch, C., Fordyce, C. S., Koehn, P.,

Monz, C., and Schroeder, J. (2008). Further meta-evaluation of machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 70–106, Columbus, Ohio. Association for Computational Linguistics.

[Callison-Burch et al., 2009] Callison-Burch, C., Koehn, P., Monz, C., and Schroeder, J. (2009). Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 1–28, Athens, Greece. Association for Computational Linguistics.

[Cherry and Lin, 2006] Cherry, C. and Lin, D. (2006). Soft syntactic constraints for word alignment through discriminative training. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 105–112, Sydney, Australia. Association for Computational Linguistics.

[Chiang, 2005] Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 263–270, Ann Arbor, Michigan. Association for Computational Linguistics.

[Costa-jussà et al., 2007] Costa-jussà, M. R., Crego, J. M., Vilar, D., Fonollosa, J. A. R., Mariño, J. B., and Ney, H. (2007). Analysis and system combination of phrase- and N-gram-based statistical machine translation systems. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 137–140, Rochester, New York. Association for Computational Linguistics.

[Eck et al., 2007a] Eck, M., Vogel, S., and Waibel, A. (2007a). Estimating phrase pair relevance for translation model pruning. In *Proceedings of the MT Summit XI.*

[Eck et al., 2007b] Eck, M., Vogel, S., and Waibel, A. (2007b). Translation model pruning via usage statistics for statistical machine translation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 21–24, Rochester, New York. Association for Computational Linguistics.

[Fossum et al., 2008] Fossum, V. L., Knight, K., and Abney, S. (2008). Using syntax to improve word alignment precision for syntax-based machine

translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 44–52, Columbus, Ohio. Association for Computational Linguistics.

[Foster et al., 2006] Foster, G., Kuhn, R., and Johnson, H. (2006). Phrasetable smoothing for statistical machine translation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 53–61, Sydney, Australia. Association for Computational Linguistics.

[Fraser and Marcu, 2007] Fraser, A. and Marcu, D. (2007). Measuring word alignment quality for statistical machine translation. *Computational Linguistics, Squibs & Discussion*, 3(33):293–303.

[Ittycheriah and Roukos, 2005] Ittycheriah, A. and Roukos, S. (2005). A maximum entropy word aligner for Arabic-English machine translation. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 89–96, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

[Johnson et al., 2007] Johnson, H., Martin, J., Foster, G., and Kuhn, R. (2007). Improving translation quality by discarding most of the phrasetable. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 967–975.

[Jones et al., 2006] Jones, D. A., Anderson, T., Atwell, S., Delaney, B., Dirgin, J., Emots, M., Granoein, N., Herzog, M., Hunter, T., Jabri, S., Shen, W., and Sottung, J. (2006). Toward an interagency language roundtable based assessment of speech-to-speech translation capabilities. In *5th Conference of the Association for Machine Translation in the Americas (AMTA)*, Boston, Massachusetts.

[Knight, 1999] Knight, K. (1999). Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4):607–615.

[Koehn and Haddow, 2009] Koehn, P. and Haddow, B. (2009). Interactive assistance to human translators using statistical machine translation methods. In *Proceedings of the MT Summit XII*.

[Koehn and Hoang, 2007] Koehn, P. and Hoang, H. (2007). Factored translation models. In *Proceedings of the 2007 Joint Conference on Empiri-*

cal Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pages 868–876.

[Kutsumi et al., 2005] Kutsumi, T., Yoshimi, T., Kotani, K., Sata, I., and Isahara, H. (2005). Selection of entries for a bilingual dictionary from aligned translation equivalents using support vector machines. In *Proceedings of the Tenth Machine Translation Summit (MT Summit X)*, Phuket, Thailand.

[Ma et al., 2008] Ma, Y., Ozdowska, S., Sun, Y., and Way, A. (2008). Improving word alignment using syntactic dependencies. In *Proceedings of the ACL-08: HLT Second Workshop on Syntax and Structure in Statistical Translation (SSST-2)*, pages 69–77, Columbus, Ohio. Association for Computational Linguistics.

[Mariño et al., 2006] Mariño, J. B., Banchs, R. E., Crego, J. M., de Gispert, A., Lambert, P., Fonollosa, J. A. R., and Costa-jussà, M. R. (2006). N-gram-based machine translation. *Computational Linguistics*, 32(4).

[Matusov et al., 2004] Matusov, E., Zens, R., and Ney, H. (2004). Symmetric word alignments for statistical machine translation. In *Proceedings of Coling 2004*, pages 219–225, Geneva, Switzerland. COLING.

[Miller et al., 1993] Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. J. (1993). Introduction to WordNet: An online lexical database. Technical Report CSL 43, Cognitive Science Laboratory Princeton University.

[Moore, 2005] Moore, R. C. (2005). A discriminative framework for bilingual word alignment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 81–88, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

[Moore et al., 2006] Moore, R. C., Yih, W.-t., and Bode, A. (2006). Improved discriminative bilingual word alignment. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 513–520, Sydney, Australia. Association for Computational Linguistics.

[Niehues and Vogel, 2008] Niehues, J. and Vogel, S. (2008). Discriminative word alignment via alignment matrix modeling. In *Proceedings of*

the Third Workshop on Statistical Machine Translation, pages 18–25, Columbus, Ohio. Association for Computational Linguistics.

[Och, 2003] Och, F. J. (2003). Minimum error rate training in statistical machine translation. In Hinrichs, E. and Roth, D., editors, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

[Och and Ney, 2003] Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1).

[Papineni et al., 2002] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics (ACL)*.

[Przybocki et al., 2008] Przybocki, M., Peterson, K., and Bronsart, S. (2008). Official results of the nist 2008 "metrics for machine translation" challenge (metricsmatr08).

[Quirk and Menezes, 2006] Quirk, C. and Menezes, A. (2006). Do we need phrases? challenging the conventional wisdom in statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 9–16, New York City, USA. Association for Computational Linguistics.

[Ren et al., 2007] Ren, D., Wu, H., and Wang, H. (2007). Improving statistical word alignment with various clues. In *Proceedings of the MT Summit XI*.

[Taskar et al., 2005] Taskar, B., Simon, L.-J., and Klein, D. (2005). A discriminative matching approach to word alignment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 73–80, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

[Truss, 2003] Truss, L. (2003). *Eats, Shoots & Leaves — The Zero Tolerance Approach to Punctuation*. Profile Books.

[Venkatapathy and Joshi, 2007] Venkatapathy, S. and Joshi, A. (2007). Discriminative word alignment by learning the alignment structure and syntactic divergence between a language pair. In *Proceedings of SSST,*

NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation, pages 49–56, Rochester, New York. Association for Computational Linguistics.

[Vogel et al., 1996] Vogel, S., Ney, H., and Tillmann, C. (1996). HMM-based word alignment in statistical translation. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*.

[Wu and Wang, 2005] Wu, H. and Wang, H. (2005). Boosting statistical word alignment. In *Proceedings of the Tenth Machine Translation Summit (MT Summit X)*, Phuket, Thailand.

[Wu and Wang, 2007] Wu, H. and Wang, H. (2007). Comparative study of word alignment heuristics and phrase-based SMT. In *Proceedings of the MT Summit XI*.

[Wu et al., 2006] Wu, H., Wang, H., and Liu, Z. (2006). Boosting statistical word alignment using labeled and unlabeled data. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 913–920, Sydney, Australia. Association for Computational Linguistics.

[Zens and Ney, 2007] Zens, R. and Ney, H. (2007). Efficient phrase-table representation for machine translation with applications to online MT and speech translation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 492–499, Rochester, New York. Association for Computational Linguistics.

[Zettlemoyer and Moore, 2007] Zettlemoyer, L. and Moore, R. C. (2007). Selective phrase pair extraction for improved statistical machine translation. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 209–212, Rochester, New York. Association for Computational Linguistics.