Chapter 25
Machine Translation

The process of translating comprises in its essence the whole secret of human understanding and social communication...

Attributed to Hans-Georg Gadamer

What is translation? On a platter
A poet's pale and glaring head,
A parrot's screech, a monkey's chatter,
And profanation of the dead.
Nabokov, *On Translating Eugene Onegin*

Proper words in proper places
Jonathan Swift

Machine translation
MT

This chapter introduces techniques for **machine translation** (**MT**), the use of computers to automate some or all of the process of translating from one language to another. Translation, in its full generality, is a difficult, fascinating, and intensely human endeavor, as rich as any other area of human creativity. Consider the following passage from the end of Chapter 45 of the 18th-century novel *The Story of the Stone*, also called *Dream of the Red Chamber*, by Cao Xue Qin (Cao, 1792), transcribed in the Mandarin dialect:

> dai yu zi zai chuang shang gan nian bao chai... you ting jian chuang wai zhu shao xiang ye zhe shang, yu sheng xi li, qing han tou mu, bu jue you di xia lei lai.

Fig. 25.1 shows the English translation of this passage by David Hawkes, in sentences labeled E_1-E_4. For ease of reading, instead of giving the Chinese, we have shown the English glosses of each Chinese word IN SMALL CAPS. Words in blue are Chinese words not translated into English, or English words not in the Chinese. We have shown **alignment** lines between words that roughly correspond in the two languages.

Consider some of the issues involved in this translation. First, the English and Chinese texts are very different structurally and lexically. The four English sentences (notice the periods in blue) correspond to one long Chinese sentence. The word order of the two texts is very different, as we can see by the many crossed alignment lines in Fig. 25.1. The English has many more words than the Chinese, as we can see by the large number of English words marked in blue. Many of these differences are caused by structural differences between the two languages. For example, because Chinese rarely marks verbal aspect or tense, the English translation has additional words like *as*, *turned to*, and *had begun*, and Hawkes had to decide to translate Chinese *tou* as *penetrated*, rather than say *was penetrating* or *had penetrated*. Chinese has fewer articles than English, explaining the large number of blue *the*s. Chinese also uses far

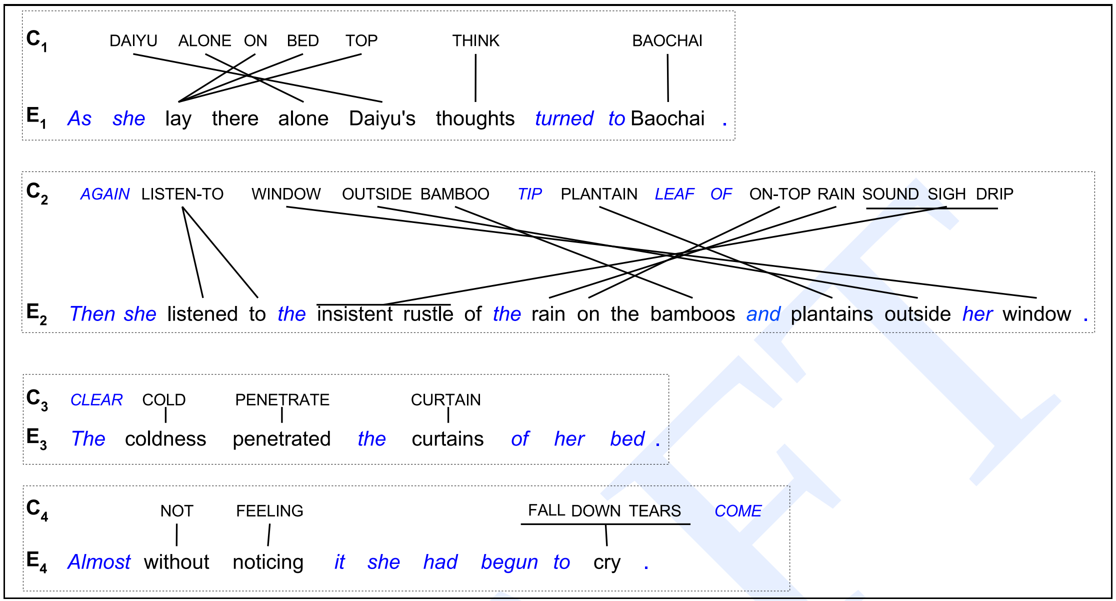

C1 DAIYU ALONE ON BED TOP THINK BAOCHAI
E1 As she lay there alone Daiyu's thoughts turned to Baochai .
C2 AGAIN LISTEN-TO WINDOW OUTSIDE BAMBOO TIP PLANTAIN LEAF OF ON-TOP RAIN SOUND SIGH DRIP
E2 Then she listened to the insistent rustle of the rain on the bamboos and plantains outside her window .
C3 CLEAR COLD PENETRATE CURTAIN
E3 The coldness penetrated the curtains of her bed .
C4 NOT FEELING FALL DOWN TEARS COME
E4 Almost without noticing it she had begun to cry .

Figure 25.1 A Chinese passage from *Dream of the Red Chamber*, with the Chinese words represented by English glosses IN SMALL CAPS. Alignment lines are drawn between 'Chinese' words and their English translations. Words in italics are Chinese words not translated into English, or English words not in the original Chinese.

fewer pronouns than English, so Hawkes had to insert *she* and *her* in many places into the English translation.

Stylistic and cultural differences are another source of difficulty for the translator. Unlike English names, Chinese names are made up of regular content words with meanings. Hawkes chose to use transliterations (*Daiyu*) for the names of the main characters but to translate names of servants by their meanings (Aroma, Skybright). To make the image clear for English readers unfamiliar with Chinese bed-curtains, Hawkes translated *ma* ('curtain') as *curtains of her bed*. The phrase *bamboo tip plantain leaf*, although elegant in Chinese, where such four-character phrases are a hallmark of literate prose, would be awkward if translated word-for-word into English, and so Hawkes used simply *bamboos and plantains*.

Translation of this sort clearly requires a deep and rich understanding of the source language and the input text, and a sophisticated, poetic, and creative command of the target language. The problem of automatically performing high-quality literary translation between languages as different as Chinese to English is thus far too hard to automate completely.

However, even non-literary translations between such similar languages as English and French can be difficult. Here is an English sentence from the Hansards corpus of Canadian parliamentary proceedings, with its French translation:

> **English**: Following a two-year transitional period, the new Foodstuffs Ordinance for Mineral Water came into effect on April 1, 1988. Specifically, it contains more stringent requirements regarding quality consistency and purity guarantees.
> **French:** La nouvelle ordonnance fèdèrale sur les denrées alimentaires concernant entre autres les eaux minérales, entrée en vigueur le ler avril 1988 aprés une période transitoire

de deux ans. exige surtout une plus grande constance dans la qualité et une garantie de la pureté.
French gloss: THE NEW ORDINANCE FEDERAL ON THE STUFF FOOD CONCERNING AMONG OTHERS THE WATERS MINERAL CAME INTO EFFECT THE 1ST APRIL 1988 AFTER A PERIOD TRANSITORY OF TWO YEARS REQUIRES ABOVE ALL A LARGER CONSISTENCY IN THE QUALITY AND A GUARANTEE OF THE PURITY.

Despite the strong structural and vocabulary overlaps between English and French, such translation, like literary translation, still has to deal with differences in word order (e.g., the location of the *following a two-year transitional period* phrase) and in structure (e.g., English uses the noun *requirements* while the French uses the verb *exige* 'REQUIRE').

Nonetheless, such translations are much easier, and a number of non-literary translation tasks can be addressed with current computational models of machine translation, including: (1) tasks for which a **rough translation** is adequate, (2) tasks where a human **post-editor** is used, and (3) tasks limited to small **sublanguage** domains in which **fully automatic high quality translation** (**FAHQT**) is still achievable.

FAHQT

Information acquisition on the web is the kind of task where a rough translation may still be useful. Suppose you were at the market this morning and saw some lovely *plátanos* (plantains, a kind of banana) at the local Caribbean grocery store and you want to know how to cook them. You go to the web, and find the following recipe:

Platano en Naranja
Para 6 personas
3 Plátanos maduros
1 taza de jugo (zumo) de naranja
1/8 cucharadita de nuez moscada en polvo
2 cucharadas de mantequilla derretida
5 cucharadas de azúcar morena o blanc
1 cucharada de ralladura de naranja
1 cucharada de canela en polvo (opcional)
Pelar los plátanos, cortarlos por la mitad y, luego, a lo largo. Engrasar una fuente o pirex con margarina. Colocar los plátanos y bañarlos con la mantequilla derretida. En un recipiente hondo, mezclar el jugo (zumo) de naranja con el azúcar, jengibre, nuez moscada y ralladura de naranja. Verter sobre los plátanos y hornear a 325 ° F. Los primeros 15 minutos, dejar los pátanos cubiertos, hornear 10 o 15 minutos más destapando los plátanos

An MT engine produces the following translation:

Platano in Orange
For 6 people
3 Bananas mature
1 cup juice (juice) orange
1/8 teaspoon nutmeg powder
2 tablespoon melted butter
5 tablespoons brown sugar or white
1 tablespoon ralladura orange
1 tablespoon cinnamon powder (optional)
Peel bananas, cut in half and then along. Grease a source or pirex with margarine. Put bananas and showering them with the melted butter. In a deep bowl, mix the juice (juice) orange with the sugar, ginger, nutmeg and ralladura orange. Pour over bananas and bake to 350° F. The first 15 minutes, leave covered bananas, bake 10 to 15 minutes more uncovering bananas.

While there are still lots of confusions in this translation (is it for bananas or plantains? What exactly is the pot we should use? What is *ralladura*?) it's probably enough, perhaps after looking up one or two words, to get a basic idea of something to try in the kitchen with your new purchase!

An MT system can also be used to speed-up the human translation process, by pro-

Post-editing

computer-aided human translation

localization

ducing a draft translation that is fixed up in a **post-editing** phase by a human translator. Strictly speaking, systems used in this way are doing **computer-aided human translation** (CAHT or CAT) rather than (fully automatic) machine translation. This model of MT usage is effective especially for high volume jobs and those requiring quick turn-around, such as the translation of software manuals for **localization** to reach new markets.

sublanguage

Weather forecasting is an example of a **sublanguage** domain that can be modeled completely enough to use raw MT output even without post-editing. Weather forecasts consist of phrases like *Cloudy with a chance of showers today and Thursday*, or *Outlook for Friday: Sunny*. This domain has a limited vocabulary and only a few basic phrase types. Ambiguity is rare, and the senses of ambiguous words are easily disambiguated based on local context, using word classes and semantic features such as WEEKDAY, PLACE, or TIME POINT. Other domains that are sublanguage-like include equipment maintenance manuals, air travel queries, appointment scheduling, and restaurant recommendations.

Applications for machine translation can also be characterized by the number and direction of the translations. Localization tasks like translations of computer manuals require one-to-many translation (from English into many languages). One-to-many translation is also needed for non-English speakers around the world to access web information in English. Conversely, many-to-one translation (into English) is relevant for anglophone readers who need the gist of web content written in other languages. Many-to-many translation is relevant for environments like the European Union, where 23 official languages (at the time of this writing) need to be intertranslated.

Before we turn to MT systems, we begin in section 25.1 by summarizing key differences among languages. The three classic models for doing MT are then presented in Sec. 25.2: the **direct**, **transfer**, and **interlingua** approaches. We then investigate in detail modern **statistical MT** in Secs. 25.3-25.8, finishing in Sec. 25.9 with a discussion of **evaluation**.

25.1 Why is Machine Translation So Hard?

Translation divergence

We began this chapter with some of the issues that made it hard to translate *The Story of the Stone* from Chinese to English. In this section we look in more detail about what makes translation difficult. We'll discuss what makes languages similar or different, including **systematic** differences that we can model in a general way, as well as **idiosyncratic** and lexical differences that must be dealt with one by one. These differences between languages are referred to as **translation divergences** and an understanding of what causes them will help us in building models that overcome the differences (Dorr, 1994).

25.1.1 Typology

When you accidentally pick up a radio program in some foreign language it seems like chaos, completely unlike the familiar languages of your everyday life. But there are

Universal

patterns in this chaos, and indeed, some aspects of human language seem to be **universal**, holding true for every language. Many universals arise from the functional role of language as a communicative system by humans. Every language, for example, seems to have words for referring to people, for talking about women, men, and children, eating and drinking, for being polite or not. Other universals are more subtle; for example Ch. 5 mentioned that every language seems to have nouns and verbs.

Typology

Even when languages differ, these differences often have systematic structure. The study of systematic cross-linguistic similarities and differences is called **typology** (Croft (1990), Comrie (1989)). This section sketches some typological facts about crosslinguistic similarity and difference.

Isolating

Polysynthetic

Agglutinative

Fusion

Morphologically, languages are often characterized along two dimensions of variation. The first is the number of morphemes per word, ranging from **isolating** languages like Vietnamese and Cantonese, in which each word generally has one morpheme, to **polysynthetic** languages like Siberian Yupik (“Eskimo”), in which a single word may have very many morphemes, corresponding to a whole sentence in English. The second dimension is the degree to which morphemes are segmentable, ranging from **agglutinative** languages like Turkish (discussed in Ch. 3), in which morphemes have relatively clean boundaries, to **fusion** languages like Russian, in which a single affix may conflate multiple morphemes, like *-om* in the word *stolom*, (table-SG-INSTR-DECL1) which fuses the distinct morphological categories instrumental, singular, and first declension.

SVO

SOV

VSO

Syntactically, languages are perhaps most saliently different in the basic word order of verbs, subjects, and objects in simple declarative clauses. German, French, English, and Mandarin, for example, are all **SVO** (**Subject-Verb-Object**) languages, meaning that the verb tends to come between the subject and object. Hindi and Japanese, by contrast, are **SOV** languages, meaning that the verb tends to come at the end of basic clauses, while Irish, Arabic, and Biblical Hebrew are **VSO** languages. Two languages that share their basic word-order type often have other similarities. For example **SVO** languages generally have **prepositions** while **SOV** languages generally have **postpositions**.

For example in the following SVO English sentence, the verb *adores* is followed by its argument VP *listening to music*, the verb *listening* is followed by its argument PP *to music*, and the preposition *to* is followed by its argument *music*. By contrast, in the Japanese example which follows, each of these orderings is reversed; both verbs are *preceded* by their arguments, and the postposition follows its argument.

(25.1) English: *He adores listening to music*

Japanese:	*kare*	*ha*	*ongaku*	*wo*	*kiku*	*no*	*ga*	*daisuki desu*
	he		music	to	listening			adores

Head-marking

Another important dimension of typological variation has to do with **argument structure** and **linking** of predicates with their arguments, such as the difference between **head-marking** and **dependent-marking** languages (Nichols, 1986). Head-marking languages tend to mark the relation between the head and its dependents on the head. Dependent-marking languages tend to mark the relation on the non-head. Hungarian, for example, marks the possessive relation with an affix (A) on the head noun (H), where English marks it on the (non-head) possessor:

(25.2) English: *the man-*^{A}*'s* ^{H}*house*
Hungarian: *az ember* ^{H}*ház-*^{A}*a*
the man house-his

Typological variation in linking can also relate to how the conceptual properties of an event are mapped onto specific words. Talmy (1985) and (1991) noted that languages can be characterized by whether direction of motion and manner of motion are marked on the verb or on the "satellites": particles, prepositional phrases, or adverbial phrases. For example a bottle floating out of a cave would be described in English with the direction marked on the particle *out*, while in Spanish the direction would be marked on the verb:

(25.3) English: *The bottle floated out.*
Spanish: La botella salió flotando.
The bottle exited floating.

Verb-framed

Satellite-framed

Verb-framed languages mark the direction of motion on the verb (leaving the satellites to mark the manner of motion), like Spanish *acercarse* 'approach', *alcanzar* 'reach', *entrar* 'enter', *salir* 'exit'. **Satellite-framed** languages mark the direction of motion on the satellite (leaving the verb to mark the manner of motion), like English *crawl out*, *float off*, *jump down*, *walk over to*, *run after*. Languages like Japanese, Tamil, and the many languages in the Romance, Semitic, and Mayan languages families, are verb-framed; Chinese as well as non-Romance Indo-European languages like English, Swedish, Russian, Hindi, and Farsi, are satellite-framed (Talmy, 1991; Slobin, 1996).

Finally, languages vary along a typological dimension related to the things they can omit. Many languages require that we use an explicit pronoun when talking about a referent that is given in the discourse. In other languages, however, we can sometimes omit pronouns altogether as the following example from Spanish shows, using the $\emptyset$-notation introduced in Ch. 21:

(25.4) [El jefe]$_{i}$ dio con un libro. $\emptyset_{i}$ Mostró a un descifrador ambulante.
[The boss] came upon a book. [He] showed it to a wandering decoder.

Pro-drop

Referential density

Cold language

Hot language

Languages which can omit pronouns in these ways are called **pro-drop** languages. Even among the pro-drop languages, there are marked differences in frequencies of omission. Japanese and Chinese, for example, tend to omit far more than Spanish. We refer to this dimension as **referential density**; languages which tend to use more pronouns are more referentially dense than those that use more zeros. Referentially sparse languages, like Chinese or Japanese, that require the hearer to do more inferential work to recover antecedents are called **cold** languages. Languages that are more explicit and make it easier for the hearer are called **hot** languages. The terms *hot* and *cold* are borrowed from Marshall McLuhan's (1964) distinction between hot media like movies, which fill in many details for the viewer, versus cold media like comics, which require the reader to do more inferential work to fill out the representation (Bickel, 2003).

Each typological dimension can cause problems when translating between languages that differ along them. Obviously translating from SVO languages like English to SOV languages like Japanese requires huge structural reorderings, since all the constituents are at different places in the sentence. Translating from a satellite-framed to a verb-framed language, or from a head-marking to a dependent-marking language, requires changes to sentence structure and constraints on word choice. Languages with extensive pro-drop, like Chinese or Japanese, cause huge problems for translation into non-pro-drop languages like English, since each zero has to be identified and the anaphor recovered.

25.1.2 Other Structural Divergences

Many structural divergences between languages are based on typological differences. Others, however, are simply idiosyncratic differences that are characteristic of particular languages or language pairs. For example in English the unmarked order in a noun-phrase has adjectives precede nouns, but in French and Spanish adjectives generally follow nouns. [1]

	Spanish	*bruja*	*verde*	**French**	*maison*	*bleue*
(25.5)		witch	green		house	blue
	English	“green witch”			“blue house”	

Chinese relative clauses are structured very differently than English relative clauses, making translation of long Chinese sentences very complex.

Language-specific constructions abound. English, for example, has an idiosyncratic syntactic construction involving the word *there* that is often used to introduce a new scene in a story, as in *there burst into the room three men with guns*. To give an idea of how trivial, yet crucial, these differences can be, think of dates. Dates not only appear in various formats — typically DD/MM/YY in British English, MM/DD/YY in American English, and YYMMDD in Japanese—but the calendars themselves may also differ. Dates in Japanese, for example, are often relative to the start of the current Emperor’s reign rather than to the start of the Christian Era.

25.1.3 Lexical Divergences

Lexical divergences also cause huge difficulties in translation. We saw in Ch. 20, for example, that the English source language word *bass* could appear in Spanish as the fish *lubina* or the instrument *bajo*. Thus translation often requires solving the exact same problems as word sense disambiguation, and the two fields are closely linked.

In English the word *bass* is homonymous; the two senses of the word are not closely related semantically, and so it is natural that we would have to disambiguate in order to translate. Even in cases of polysemy, however, we often have to disambiguate if the target language doesn’t have the exact same kind of polysemy. The English word *know*, for example, is polysemous; it can refer to knowing of a fact or proposition (*I know that snow is white*) or familiarity with a person or location (*I know Jon Stewart*). It turns out that translating these different senses requires using distinct French verbs, including the verbs *connaître*, and *savoir*. *Savoir* is generally used with sentential complements to indicate knowledge or mental representation of a fact or proposition,

[1] As always, there are exceptions to this generalization, such as *galore* in English and *gros* in French; furthermore in French some adjectives can appear before the noun with a different meaning; *route mauvaise* ‘bad road, badly-paved road’ versus *mauvaise route* ‘wrong road’ (Waugh, 1976).

or verbal complements to indicate knowledge of how to do something (e.g., WordNet 3.0 senses #1, #2, #3). *Connaître* is generally used with NP complements to indicate familiarity or acquaintance with people, entities, or locations (e.g., WordNet 3.0 senses #4, #7). Similar distinctions occur in German, Chinese, and many other languages:

(25.6) **English:** I know he just bought a book.

(25.7) **French:** Je sais qu'il vient d'acheter un livre.

(25.8) **English:** I know John.

(25.9) **French:** Je connais Jean.

The *savoir/connaître* distinction corresponds to different groups of WordNet senses. Sometimes, however, a target language will make a distinction that is not even recognized in fine-grained dictionaries. German, for example, uses two distinct words for what in English would be called a *wall*: *Wand* for walls inside a building, and *Mauer* for walls outside a building. Similarly, where English uses the word *brother* for any male sibling, both Japanese and Chinese have distinct words for *older brother* and *younger brother* (Chinese *gege* and *didi*, respectively).

In addition to these distinctions, lexical divergences can be grammatical. For example, a word may translate best to a different part-of-speech in the target language. Many English sentences involving the verb *like* must be translated into German using the adverbial *gern*; thus *she likes to sing* maps to *sie singt gerne* (SHE SINGS LIKINGLY).

In translation, we can think of sense disambiguation as a kind of **specification**; we have to make a vague word like *know* or *bass* more specific in the target language. This kind of specification is also quite common with grammatical differences. Sometimes one language places more grammatical constraints on word choice than another. French and Spanish, for example, marks gender on adjectives, so an English translation into French requires specifying adjective gender. English distinguishes gender in pronouns where Mandarin does not; thus translating a third-person singular pronoun *tā* from Mandarin to English (*he*, *she*, or *it*) requires deciding who the original referent was. In Japanese, because there is no single word for *is*, the translator must choose between *iru* or *aru*, based on whether the subject is animate or not.

The way that languages differ in lexically dividing up conceptual space may be more complex than this one-to-many translation problem, leading to many-to-many mappings. For example Fig. 25.2 summarizes some of the complexities discussed by Hutchins and Somers (1992) in relating English *leg*, *foot*, and *paw*, to the French *jambe*, *pied*, *patte*, etc.

Lexical gap

Further, one language may have a **lexical gap**, where no word or phrase, short of an explanatory footnote, can express the meaning of a word in the other language. For example, Japanese does not have a word for *privacy*, and English does not have a word for Japanese *oyakoko* or Chinese *xiáo* (we make do with the awkward phrase *filial piety* for both).

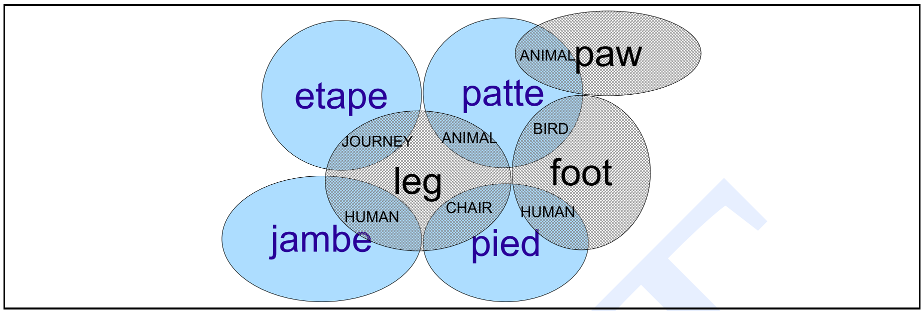


Figure 25.2 The complex overlap between English *leg*, *foot*, etc, and various French translations like *patte* discussed by Hutchins and Somers (1992) .

25.2 Classical MT & the Vauquois Triangle

The next few sections introduce the classical pre-statistical architectures for machine translation. Real systems tend to involve combinations of elements from these three architectures; thus each is best thought of as a point in an algorithmic design space rather than as an actual algorithm.

In **direct** translation, we proceed word-by-word through the source language text, translating each word as we go. Direct translation uses a large bilingual dictionary, each of whose entries is a small program with the job of translating one word. In **transfer** approaches, we first parse the input text, and then apply rules to transform the source language parse structure into a target language parse structure. We then generate the target language sentence from the parse structure. In **interlingua** approaches, we analyze the source language text into some abstract meaning representation, called an **interlingua**. We then generate into the target language from this interlingual representation.

Vauquois triangle

A common way to visualize these three approaches is with **Vauquois triangle** shown in Fig. 25.3. The triangle shows the increasing depth of analysis required (on both the analysis and generation end) as we move from the direct approach through transfer approaches, to interlingual approaches. In addition, it shows the decreasing amount of transfer knowledge needed as we move up the triangle, from huge amounts of transfer at the direct level (almost all knowledge is transfer knowledge for each word) through transfer (transfer rules only for parse trees or thematic roles) through interlingua (no specific transfer knowledge).

In the next sections we'll see how these algorithms address some of the four translation examples shown in Fig. 25.2

25.2.1 Direct Translation

Direct translation

In **direct translation**, we proceed word-by-word through the source language text, translating each word as we go. We make use of no intermediate structures, except for shallow morphological analysis; each source word is directly mapped onto some target word. Direct translation is thus based on a large bilingual dictionary; each entry in the

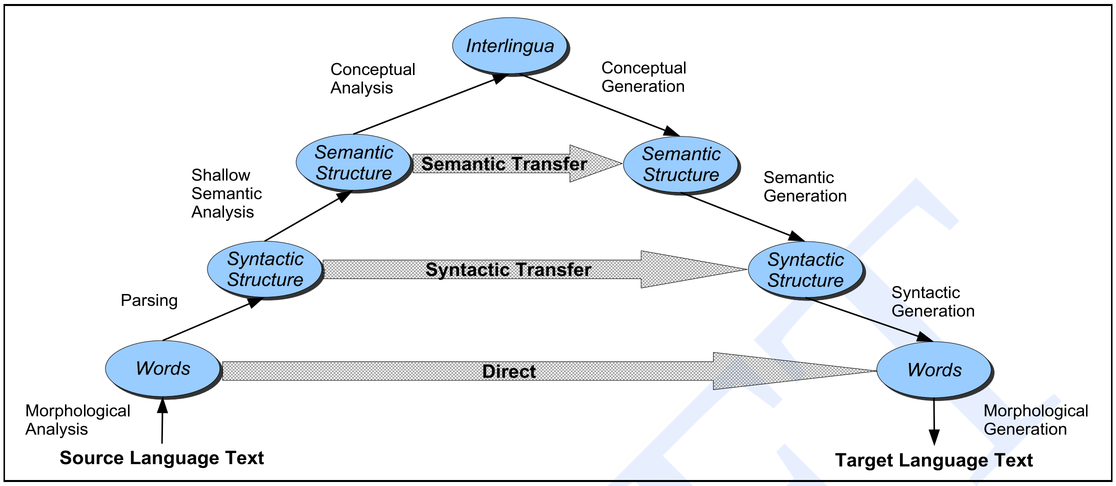


Figure 25.3 The Vauquois (1968) triangle.

English	Mary didn't slap the green witch
⇒ Spanish	*Maria no dió una bofetada a la bruja verde*
	Mary not gave a slap to the witch green
English	The green witch is at home this week
⇒ German	*Diese Woche ist die grüne Hexe zu Hause.*
	this week is the green witch at house
English	He adores listening to music
⇒ Japanese	*kare ha ongaku wo kiku no ga daisuki desu*
	he music to listening adores
Chinese	*cheng long dao xiang gang qu*
	Jackie Chan to Hong Kong go
⇒ English	Jackie Chan went to Hong Kong

Figure 25.4 Example sentences used throughout the chapter.

dictionary can be viewed as a small program whose job is to translate one word. After the words are translated, simple reordering rules can apply, for example for moving adjectives after nouns when translating from English to French.

The guiding intuition of the direct approach is that we translate by incrementally **transforming** the source language text into a target language text. While the pure direct approach is no longer used, this transformational intuition underlies all modern systems, both statistical and non-statistical.

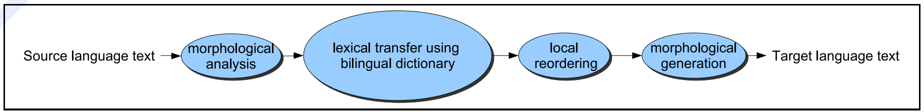


Figure 25.5 Direct machine translation. The major component, indicated by size here, is the bilingual dictionary.

Let's look at a simplified direct system on our first example, translating from En-

Input:	Mary didn't slap the green witch
After 1: Morphology	Mary DO-PAST not slap the green witch
After 2: Lexical Transfer	Maria PAST no dar una bofetada a la verde bruja
After 3: Local reordering	Maria no dar PAST una bofetada a la bruja verde
After 4: Morphology	Maria no dió una bofetada a la bruja verde

Figure 25.6 An example of processing in a direct system

glish into Spanish:

(25.10) Mary didn't slap the green witch
Maria no dió una bofetada a la bruja verde
Mary not gave a slap to the witch green

The four steps outlined in Fig. 25.5 would proceed as shown in Fig. 25.6.

Step 2 presumes that the bilingual dictionary has the phrase *dar una bofetada a* as the Spanish translation of English *slap*. The local reordering step 3 would need to switch the adjective-noun ordering from *green witch* to *bruja verde*. And some combination of ordering rules and the dictionary would deal with the negation and past tense in English *didn't*. These dictionary entries can be quite complex; a sample dictionary entry from an early direct English-Russian system is shown in Fig. 25.7.

```
function DIRECT_TRANSLATE_MUCH/MANY(word) returns Russian translation

if preceding word is how return skol'ko
else if preceding word is as return stol'ko zhe
else if word is much
  if preceding word is very return nil
  else if following word is a noun return mnogo
else /* word is many */
  if preceding word is a preposition and following word is a noun return mnogii
  else return mnogo
```

Figure 25.7 A procedure for translating *much* and *many* into Russian, adapted from Hutchins' (1986, pg. 133) discussion of Panov 1960. Note the similarity to decision list algorithms for word sense disambiguation.

While the direct approach can deal with our simple Spanish example, and can handle single-word reorderings, it has no parsing component or indeed any knowledge about phrasing or grammatical structure in the source or target language. It thus cannot reliably handle longer-distance reorderings, or those involving phrases or larger structures. This can happen even in languages very similar to English, like German, where adverbs like *heute* ('today') occur in different places, and the subject (e.g., *die grüne Hexe*) can occur after the main verb, as shown in Fig. 25.8.

Similar kinds of reorderings happen between Chinese (where goal PPs often occur preverbally) and English (where goal PPs must occur postverbally), as shown in Fig. 25.9.

Finally, even more complex reorderings occur when we translate from SVO to SOV languages, as we see in the English-Japanese example from Yamada and Knight (2002):

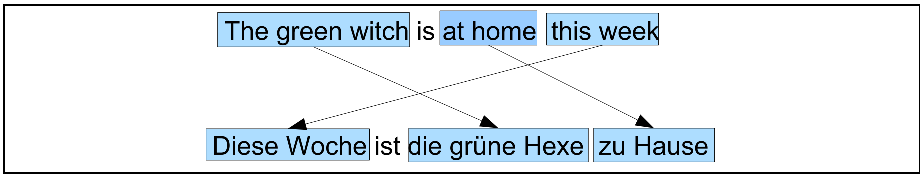


Figure 25.8 Complex reorderings necessary when translating from English to German. German often puts adverbs in initial position that English would more naturally put later. German tensed verbs often occur in second position in the sentence, causing the subject and verb to be inverted.

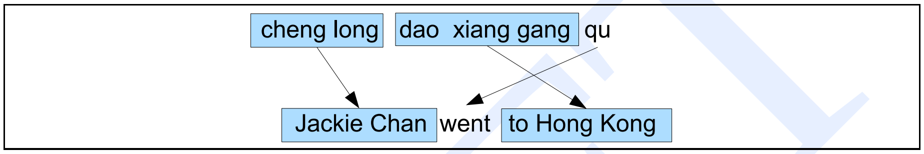


Figure 25.9 Chinese goal PPs often occur preverbally, unlike in English

.

(25.11) He adores listening to music

kare ha ongaku wo kiku no ga daisuki desu
he music to listening adores

These three examples suggest that the direct approach is too focused on individual words, and that in order to deal with real examples we'll need to add phrasal and structural knowledge into our MT models. We'll flesh out this intuition in the next section.

25.2.2 Transfer

As Sec. 25.1 illustrated, languages differ systematically in structural ways. One strategy for doing MT is to translate by a process of overcoming these differences, altering the structure of the input to make it conform to the rules of the target language. This can be done by applying **contrastive knowledge**, that is, knowledge about the differences between the two languages. Systems that use this strategy are said to be based on the **transfer model**.

Contrastive knowledge

transfer model

The transfer model presupposes a parse of the source language, and is followed by a generation phase to actually create the output sentence. Thus, on this model, MT involves three phases: **analysis**, **transfer**, and **generation**, where transfer bridges the gap between the output of the source language parser and the input to the target language generator.

It is worth noting that a parse for MT may differ from parses required for other purposes. For example, suppose we need to translate *John saw the girl with the binoculars* into French. The parser does not need to bother to figure out where the prepositional phrase attaches, because both possibilities lead to the same French sentence.

Once we have parsed the source language, we'll need rules for **syntactic transfer** and **lexical transfer**. The syntactic transfer rules will tell us how to modify the source parse tree to resemble the target parse tree.

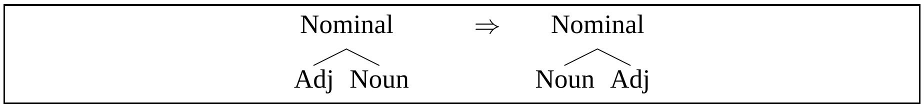


Figure 25.10 A simple transformation that reorders adjectives and nouns

syntactic transformations

Fig. 25.10 gives an intuition for simple cases like adjective-noun reordering; we transform one parse tree, suitable for describing an English phrase, into another parse tree, suitable for describing a Spanish sentence. These **syntactic transformations** are operations that map from one tree structure to another.

The transfer approach and this rule can be applied to our example *Mary did not slap the green witch*. Besides this transformation rule, we'll need to assume that the morphological processing figures out that *didn't* is composed of *do-PAST* plus *not*, and that the parser attaches the PAST feature onto the VP. Lexical transfer, via lookup in the bilingual dictionary, will then remove *do*, change *not* to *no*, and turn *slap* into the phrase *dar una bofetada a*, with a slight rearrangement of the parse tree, as suggested in Fig. 25.11.

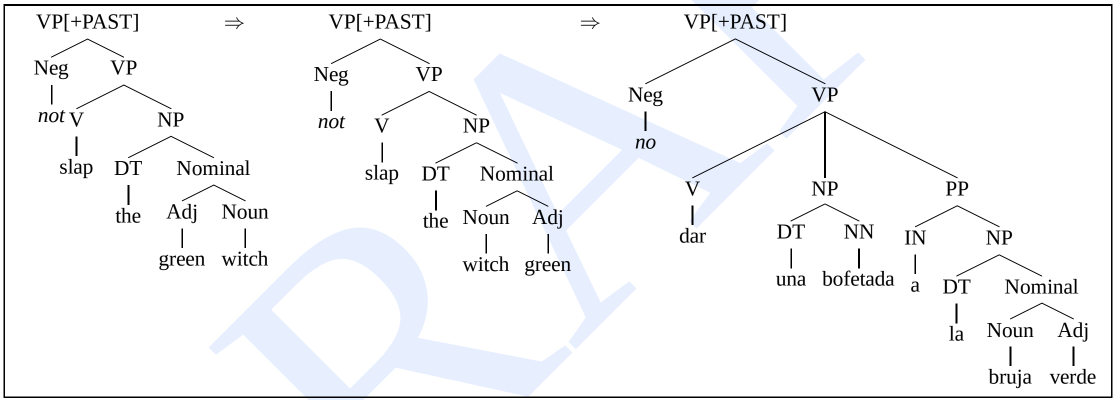


Figure 25.11 A further sketch of the transfer approach.

For translating from SVO languages like English to SOV languages like Japanese, we'll need even more complex transformations, for moving the verb to the end, changing prepositions into postpositions, and so on. An example of the result of such rules is shown in Fig. 25.12. An informal sketch of some transfer rules is shown in Fig. 25.13.

Transfer systems can be based on richer structures than just pure syntactic parses. For example a transfer based system for translating Chinese to English might have rules to deal with the fact shown in Fig. 25.9 that in Chinese PPs that fill the semantic role GOAL (like *to the store* in *I went to the store*) tend to appear before the verb, while in English these goal PPs must appear after the verb. In order to build a transformation to deal with this and related PP ordering differences, the parse of the Chinese must including thematic structure, so as to distinguish BENEFACTIVE PPs (which must occur before the verb) from DIRECTION and LOCATIVE PPs (which preferentially occur before the verb) from RECIPIENT PPs (which occur after) (Li and Thompson, 1981). We discussed how to do this kind of semantic role labeling in Ch. 20. Using semantic

Semantic transfer

roles in this way is generally called **semantic transfer**; a simple such transformation

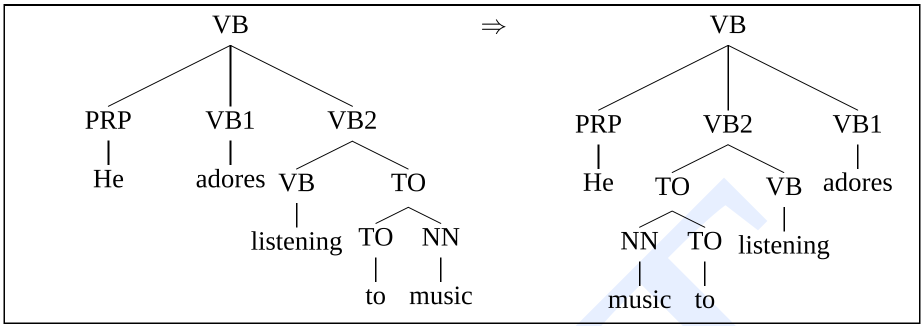


Figure 25.12 The result of Yamada and Knight's (2001) syntactic transformations from English order (SVO) to Japanese order (SOV) for the sentence *He adores listening to music* (*kare ha ongaku wo kiku no ga daisuki desu*). This transform would require rules for moving verbs after their NP and VP complements, and changing prepositions to postpositions.

	English to Spanish:		
1.	NP → Adjective$_1$ Noun$_2$	⇒	NP → Noun$_2$ Adjective$_1$
	Chinese to English:		
2.	VP → PP[+Goal] V	⇒	VP → V PP[+Goal]
	English to Japanese:		
3.	VP → V NP	⇒	VP → NP V
4.	PP → P NP	⇒	PP → NP P
5.	NP → NP$_1$ Rel. Clause$_2$	⇒	NP → Rel. Clause$_2$ NP$_1$

Figure 25.13 An informal description of some transformations.

is shown in Fig. 25.13.

In addition to syntactic transformations, transfer-based systems need to have lexical transfer rules. Lexical transfer is generally based on a bilingual dictionary, just as for direct MT. The dictionary itself can also be used to deal with problems of lexical ambiguity. For example the English word *home* has many possible translations in German, including *nach Hause* (in the sense of *going home*) *Heim* (in the sense of a *home game*), *Heimat* (in the sense of *homeland*, *home country*, or *spiritual home*), and *zu Hause* (in the sense of being *at home*). In this case, the phrase *at home* is very likely to be translated *zu Hause*, and so the bilingual dictionary can list this translation idiomatically.

Many cases of lexical transfer are too complex to deal with via a phrasal dictionary. In these cases transfer systems can do disambiguation during the source language analysis, by applying the sense disambiguation techniques of Ch. 20.

25.2.3 Combining direct and tranfer approaches in classic MT

Although the transfer metaphor offers the ability to deal with more complex source language phenomena than the direct approach, it turns out the simple SVO → SOV rules we've described above are not sufficient. In practice, we need messy rules which combine rich lexical knowledge of both languages with syntactic and semantic features. We briefly saw an example of such a rule for changing *slap* to *dar una bofetada a*.

For this reason, commercial MT systems tend to be combinations of the direct and transfer approaches, using rich bilingual dictionaries, but also using taggers and parsers. The Systran system, for example, as described in Hutchins and Somers (1992), Senellart et al. (2001), has three components. First is a shallow **analysis** stage, including:

- morphological analysis and part-of-speech tagging
- chunking of NPs, PPs, and larger phrases
- shallow dependency parsing (subjects, passives, head-modifiers)

Next is a **transfer** phase, including:

- translation of idioms,
- word sense disambiguation
- assigning prepositions based on governing verbs

Finally, in the **synthesis** stage, the system:

- applies a rich bilingual dictionary to do lexical translation
- deals with reorderings
- performs morphological generation

Thus like the direct system, the Systran system relies for much of its processing on the bilingual dictionary, which has lexical, syntactic, and semantic knowledge. Also like a direct system, Systran does reordering in a post-processing step. But like a transfer system, many of the steps are informed by syntactic and shallow semantic processing of the source language.

25.2.4 The Interlingua Idea: Using Meaning

One problem with the transfer model is that it requires a distinct set of transfer rules for each pair of languages. This is clearly suboptimal for translation systems employed in many-to-many multilingual environments like the European Union.

This suggests a different perspective on the nature of translation. Instead of directly transforming the words of the source language sentence into the target language, the interlingua intuition is to treat translation as a process of extracting the meaning of the input and then expressing that meaning in the target language. If this could be done, an MT system could do without contrastive knowledge, merely relying on the same syntactic and semantic rules used by a standard interpreter and generator for the language. The amount of knowledge needed would then be proportional to the number of languages the system handles, rather than to the square.

Interlingua

This scheme presupposes the existence of a meaning representation, or **interlingua**, in a language-independent canonical form, like the semantic representations we saw in Ch. 17. The idea is for the interlingua to represent all sentences that mean the "same" thing in the same way, regardless of the language they happen to be in. Translation in this model proceeds by performing a deep semantic analysis on the input from language X into the interlingual representation and generating from the interlingua to language Y.

What kind of representation scheme can we use as an interlingua? The predicate calculus, or a variant such as minimal recursion semantics, is one possibility. Semantic

$$\begin{bmatrix} \textsc{event} & \textsc{slapping} \\ \textsc{agent} & \textsc{Mary} \\ \textsc{tense} & \textsc{past} \\ \textsc{polarity} & \textsc{negative} \\ \textsc{theme} & \begin{bmatrix} \textsc{witch} & \\ \textsc{definiteness} & \textsc{def} \\ \textsc{attributes} & \begin{bmatrix} \textsc{has-color} & \textsc{green} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure 25.14 Interlingual representation of *Mary did not slap the green witch*.

decomposition into some kind of atomic semantic primitives is another. We will illustrate a third common approach, a simple event-based representation, in which events are linked to their arguments via a small fixed set of thematic roles. Whether we use logics or other representations of events, we'll need to specify temporal and aspectual properties of the events, and we'll also need to represent non-eventive relationships between entities, such as the *has-color* relation between *green* and *witch*. Fig. 25.2.4 shows a possible interlingual representation for *Mary did not slap the green witch* as a unification-style feature structure.

We can create these interlingual representation from the source language text using the **semantic analyzer** techniques of Ch. 18 and Ch. 20; using a semantic role labeler to discover the AGENT relation between *Mary* and the *slap* event, or the THEME relation between the *witch* and the *slap* event. We would also need to do disambiguation of the noun-modifier relation to recognize that the relationship between *green* and *witch* is the *has-color* relation, and we'll need to discover that this event has negative polarity (from the word *didn't*). The interlingua thus requires more analysis work than the transfer model, which only required syntactic parsing (or at most shallow thematic role labeling). But generation can now proceed directly from the interlingua with no need for syntactic transformations.

In addition to doing without syntactic transformations, the interlingual system does without lexical transfer rules. Recall our earlier problem of whether to translate *know* into French as *savoir* or *connaître*. Most of the processing involved in making this decision is not specific to the goal of translating into French; German, Spanish, and Chinese all make similar distinctions, and furthermore the disambiguation of *know* into concepts such as HAVE-A-PROPOSITION-IN-MEMORY and BE-ACQUAINTED-WITH-ENTITY is also important for other NLU applications that require word-senses. Thus by using such concepts in an interlingua, a larger part of the translation process can be done with general language processing techniques and modules, and the processing specific to the English-to-French translation task can be eliminated or at least reduced, as suggested in Fig. 25.3.

The interlingual model has its own problems. For example, in order to translate from Japanese to Chinese the universal interlingua must include concepts such as ELDER-BROTHER and YOUNGER-BROTHER. Using these same concepts translating from German-to-English would then require large amounts of unnecessary disambiguation. Furthermore, doing the extra work involved by the interlingua commitment requires exhaustive analysis of the semantics of the domain and formalization into an ontology. Generally this is only possible in relatively simple domains based on a database model, as in the air travel, hotel reservation, or restaurant recommendation domains, where the database definition determines the possible entities and relations. For these reasons, interlingual systems are generally only used in sublanguage domains.

25.3 Statistical MT

The three classic architectures for MT (direct, transfer, and interlingua) all provide answers to the questions of what representations to use and what steps to perform to translate. But there is another way to approach the problem of translation: to focus on the result, not the process. Taking this perspective, let's consider what it means for a sentence to be a translation of some other sentence.

This is an issue to which philosophers of translation have given a lot of thought. The consensus seems to be, sadly, that it is impossible for a sentence in one language to be a translation of a sentence in other, strictly speaking. For example, one cannot really translate Hebrew *adonai roi* ('the Lord is my shepherd') into the language of a culture that has no sheep. On the one hand, we can write something that is clear in the target language, at some cost in fidelity to the original, something like *the Lord will look after me*. On the other hand, we can be faithful to the original, at the cost of producing something obscure to the target language readers, perhaps like *the Lord is for me like somebody who looks after animals with cotton-like hair*. As another example, if we translate the Japanese phrase *fukaku hansei shite orimasu*, as *we apologize*, we are not being faithful to the meaning of the original, but if we produce *we are deeply reflecting (on our past behavior, and what we did wrong, and how to avoid the problem next time)*, then our output is unclear or awkward. Problems such as these arise not only for culture-specific concepts, but whenever one language uses a metaphor, a construction, a word, or a tense without an exact parallel in the other language.

So, true translation, which is both faithful to the source language and natural as an utterance in the target language, is sometimes impossible. If you are going to go ahead and produce a translation anyway, you have to compromise. This is exactly what translators do in practice: they produce translations that do tolerably well on both criteria.

This provides us with a hint for how to do MT. We can model the goal of translation as the production of an output that maximizes some value function that represents the importance of both faithfulness and fluency. Statistical MT is the name for a class of approaches that do just this, by building probabilistic models of faithfulness and fluency, and then combining these models to choose the most probable translation. If we chose the product of faithfulness and fluency as our quality metric, we could model the translation from a source language sentence S to a target language sentence $\hat{T}$ as:

$$\text{best-translation } \hat{T} = \text{argmax}_T \text{ faithfulness(T,S) fluency(T)}$$

This intuitive equation clearly resembles the Bayesian **noisy channel model** we've seen in Ch. 5 for spelling and Ch. 9 for speech. Let's make the analogy perfect and formalize the noisy channel model for statistical machine translation.

First of all, for the rest of this chapter, we'll assume we are translating from a foreign language sentence $F = f_1, f_2, ..., f_m$ to English. For some examples we'll use French as the foreign language, and for others Spanish. But in each case we are translating **into English** (although of course the statistical model also works for translating out of English). In a probabilistic model, the best English sentence $\hat{E} = e_1, e_2, ..., e_l$ is the one whose probability $P(E|F)$ is the highest. As is usual in the noisy channel model, we can rewrite this via Bayes rule:

$$\begin{aligned} \hat{E} &= argmax_E P(E|F) \\ &= argmax_E \frac{P(F|E)P(E)}{P(F)} \\ &= argmax_E P(F|E)P(E) \qquad (25.12) \end{aligned}$$

We can ignore the denominator $P(F)$ inside the argmax since we are choosing the best English sentence for a fixed foreign sentence F, and hence $P(F)$ is a constant. The resulting noisy channel equation shows that we need two components: a **translation model** $P(F|E)$, and a **language model** $P(E)$.

Translation model
Language model

$$\hat{E} = \underset{E \in \text{English}}{\text{argmax}} \overbrace{P(F|E)}^{\text{translation model}} \overbrace{P(E)}^{\text{language model}} \qquad (25.13)$$

Notice that applying the noisy channel model to machine translation requires that we think of things backwards, as shown in Fig. 25.15. We pretend that the foreign (source language) input F we must translate is a corrupted version of some English (target language) sentence E, and that our task is to discover the hidden (target language) sentence E that generated our observation sentence F.

The noisy channel model of statistical MT thus requires three components to translate from a French sentence F to an English sentence E:

- A **language model** to compute $P(E)$
- A **translation model** to compute $P(F|E)$
- A **decoder**, which is given F and produces the most probable E

Of these three components, we have already introduced the language model $P(E)$ in Ch. 4. Statistical MT systems are based on the same N-gram language models as speech recognition and other applications. The language model component is monolingual, and so acquiring training data is relatively easy.

The next few sections will therefore concentrate on the other two components, the translation model and the decoding algorithm.

25.4 $P(F|E)$: the Phrase-Based Translation Model

The job of the translation model, given an English sentence E and a foreign sentence F, is to assign a probability that E generates F. While we can estimate these probabilities

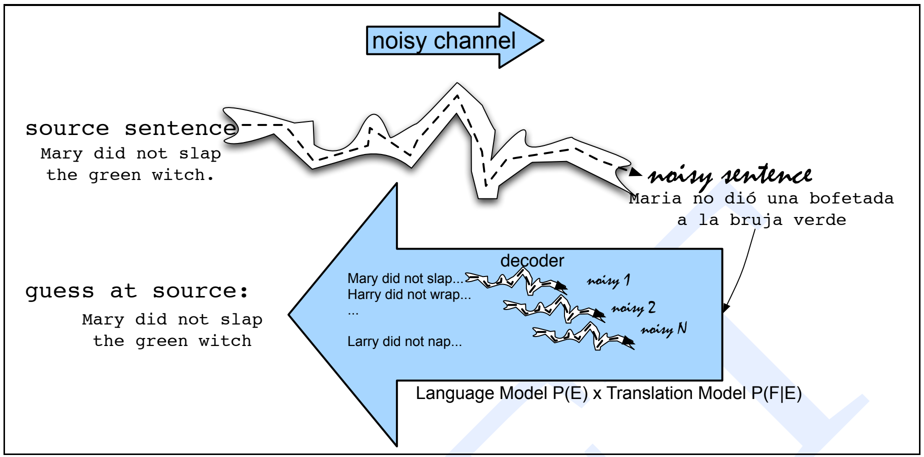


Figure 25.15 The noisy channel model of statistical MT. If we are translating a source language French to a target language English, we have to think of 'sources' and 'targets' backwards. We build a model of the generation process from an English sentence through a channel to a French sentence. Now given a French sentence to translate, we pretend it is the output of an English sentence going through the noisy channel, and search for the best possible 'source' English sentence.

by thinking about how each individual word is translated, modern statistical MT is based on the intuition that a better way to compute these probabilities is by considering the behavior of **phrases**. As we see in Fig. 25.16, repeated from page 884, entire phrases often need to be translated and moved as a unit. The intuition of **phrase-based** statistical MT is to use phrases (sequences of words) as well as single words as the fundamental units of translation.

Phrase-based

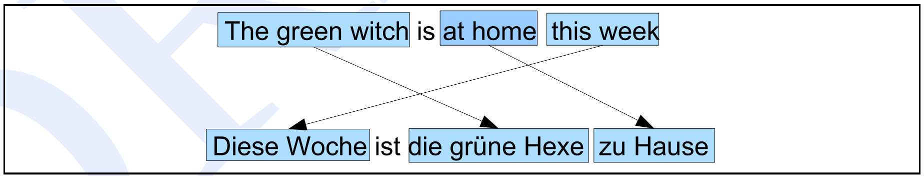


Figure 25.16 Phrasal reorderings necessary when generating German from English; repeated from Fig. 25.8.

There are a wide variety of phrase-based models; in this section we will sketch the model of Koehn et al. (2003). We'll use a Spanish example, seeing how the phrase-based model computes the probability P(*Maria no dió una bofetada a la bruja verde*|*Mary did not slap the green witch*).

The generative story of phrase-based translation has three steps. First we group the English source words into phrases $\bar{e}_1, \bar{e}_2 ... \bar{e}_I$. Next we translate each English phrase $\bar{e}_i$ into a Spanish phrase $\bar{f}_j$. Finally each of the Spanish phrases is (optionally) reordered.

The probability model for phrase-based translation relies on a **translation probability** and a **distortion probability**. The factor $\phi(\bar{f}_j|\bar{e}_i)$ is the translation probability of generating Spanish phrase $\bar{f}_j$ from English phrase $\bar{e}_i$. The reordering of the Spanish

Distortion

phrases is done by the **distortion** probability d. Distortion in statistical machine translation refers to a word having a different ('distorted') position in the Spanish sentence than it had in the English sentence; it is thus a measure of the **distance** between the positions of a phrase in the two languages. The distortion probability in phrase-based MT means the probability of two consecutive English phrases being separated in Spanish by a span (of Spanish words) of a particular length. More formally, the distortion is parameterized by $d(a_i - b_{i-1})$, where a_i is the start position of the foreign (Spanish) phrase generated by the ith English phrase $\bar{e}_i$, and b_{i-1} is the end position of the foreign (Spanish) phrase generated by the $i-1$th English phrase $\bar{e}_{i-1}$. We can use a very simple distortion probability, in which we simply raise some small constant α to the distortion. $d(a_i - b_{i-1}) = \alpha^{|a_i - b_{i-1} - 1|}$. This distortion model penalizes large distortions by giving lower and lower probability the larger the distortion.

The final translation model for phrase-based MT is:

$$P(F|E) = \prod_{i=1}^{I} \phi(\bar{f}_i, \bar{e}_i) d(a_i - b_{i-1}) \tag{25.14}$$

Let's consider the following particular set of phrases for our example sentences:[2]

Position	**1**	**2**	**3**	**4**	**5**
English	Mary	did not	slap	the	green witch
Spanish	Maria	no	dió una bofetada	a la	bruja verde

Since each phrase follows directly in order (nothing moves around in this example, unlike the German example in (25.16)) the distortions are all 1, and the probability $P(F|E)$ can be computed as:

$$\begin{aligned} P(F|E) \;=\; & P(\text{Maria}, \text{Mary}) \times d(1) \times P(\text{no}|\text{did not}) \times d(1) \times \\ & P(\text{dió una bofetada}|\text{slap}) \times d(1) \times P(\text{a la}|\text{the}) \times d(1) \times \\ & P(\text{bruja verde}|\text{green witch}) \times d(1) \end{aligned} \tag{25.15}$$

In order to use the phrase-based model, we need two more things. We need a model of **decoding**, so we can go from a surface Spanish string to a hidden English string. And we need a model of **training**, so we can learn parameters. We'll introduce the decoding algorithm in Sec. 25.8. Let's turn first to training.

How do we learn the simple phrase-based translation probability model in (25.14)? The main set of parameters that needs to be trained is the set of phrase translation probabilities $\phi(\bar{f}_i, \bar{e}_i)$.

These parameters, as well as the distortion constant α, could be set if only we had a large bilingual training set, in which each Spanish sentence was paired with an English sentence, and if furthermore we knew exactly which phrase in the Spanish sentence was translated by which phrase in the English sentence. We call such a mapping a

Phrase alignment

phrase alignment.

[2] Exactly which phrases we use depends on which phrases are discovered in the training process, as described in Sec. 25.7; thus for example if we don't see the phrase *green witch* in our training data, we would have to translate *green* and *witch* independently.

The table of phrases above showed an implicit alignment of the phrases for this sentence, for example *green witch* aligned with *bruja verde*. If we had a large training set with each pair of sentences labeled with such a phrase alignment, we could just count the number of times each phrase-pair occurred, and normalize to get probabilities:

$$\phi(\bar{f},\bar{e}) = \frac{\text{count}(\bar{f},\bar{e})}{\sum_{\bar{f}}\text{count}(\bar{f},\bar{e})} \tag{25.16}$$

Phrase translation table

We could store each phrase pair $(\bar{f},\bar{e})$, together with its probability $\phi(\bar{f},\bar{e})$, in a large **phrase translation table**.

Word alignment

Alas, we don't have large hand-labeled phrase-aligned training sets. But it turns out that we can extract phrases from another kind of alignment called a **word alignment**. A word alignment is different than a phrase alignment, because it shows exactly which Spanish word aligns to which English word inside each phrase. We can visualize a word alignment in various ways. Fig. 25.17 and Fig. 25.18 show a graphical model and an alignment matrix, respectively, for a word alignment.

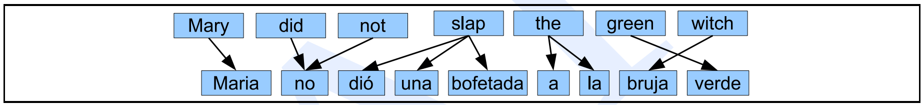
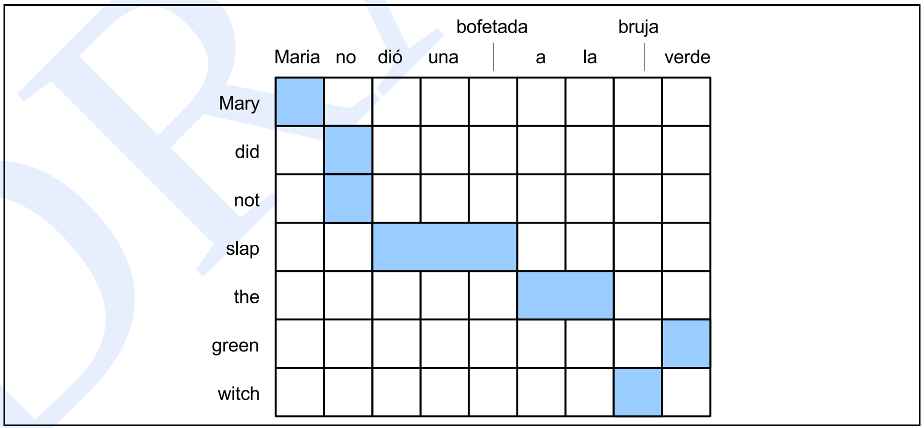


Figure 25.17 A graphical model representation of a word alignment between the English and Spanish sentences. We will see later how to extract phrases.
.



	Maria	no	dió	una	bofetada	a	la	bruja	verde
Mary	■								
did		■							
not		■							
slap			■	■	■				
the						■	■		
green									■
witch								■	

Figure 25.18 An alignment matrix representation of a word alignment between the English and Spanish sentences. We will see later how to extract phrases.
.

The next section introduces a few algorithms for deriving word alignments. We then show in Sec. 25.7 how we can extract a phrase table from word alignments, and finally in Sec. 25.8 how the phrase table can be used in decoding.

25.5 Alignment in MT

Word alignment

All statistical translation models are based on the idea of a **word alignment**. A word alignment is a mapping between the source words and the target words in a set of parallel sentences.

Fig. 25.19 shows a visualization of an alignment between the English sentence *And the program has been implemented* and the French sentence *Le programme a été mis en application*. For now, we assume that we already know which sentences in the English text aligns with which sentences in the French text.

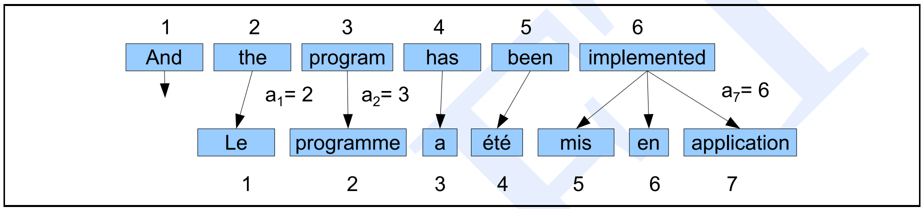


Figure 25.19 An alignment between an English and a French sentence, after Brown et al. (1993). Each French word aligns to a single English word.

In principle, we can have arbitrary alignment relationships between the English and French word. But the word alignment models we will present (IBM Models 1 and 3 and the HMM model) make a more stringent requirement, which is that each French word comes from exactly one English word; this is consistent with Fig. 25.19. One advantage of this assumption is that we can represent an alignment by giving the index number of the English word that the French word comes from. We can thus represent the alignment shown in Fig. 25.19 as $A = 2,3,4,5,6,6,6$. This is a very likely alignment. A very unlikely alignment, by contrast, might be $A = 3,3,3,3,3,3,3$.

We will make one addition to this basic alignment idea, which is to allow words to appear in the foreign sentence that don't align to any word in the English sentence. We model these words by assuming the existence of a NULL English word e_0 at position 0. Words in the foreign sentence that are not in the English sentence, called **spurious words**, may be generated by e_0. Fig. 25.20 shows the alignment of spurious Spanish *a* to English NULL.[3]

Spurious word

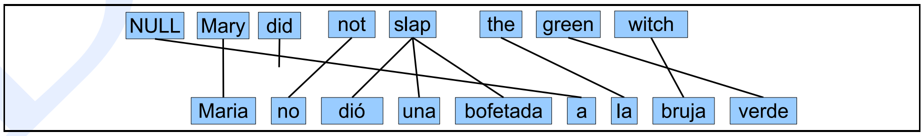


Figure 25.20 The alignment of the **spurious** Spanish word *a* to the English NULL word e_0.

While the simplified model of alignment above disallows many-to-one or many-to-many alignments, we will discuss more powerful translation models that allow such alignments. Here are two such sample alignments; in Fig. 25.21 we see an alignment

[3] While this particular *a* might instead be aligned to English *slap*, there are many cases of spurious words which have no other possible alignment site.

which is many-to-one; each French word does not align to a single English word, although each English word does align to a single French word.

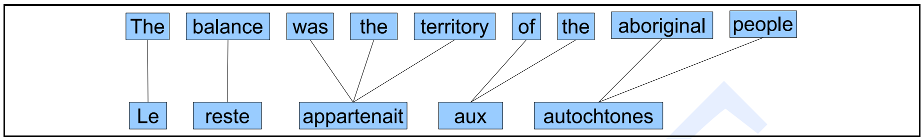


Figure 25.21 An alignment between an English and a French sentence, in which each French word does not align to a single English word, but each English word aligns to one French word. Adapted from Brown et al. (1993).

Fig. 25.22 shows an even more complex example, in which multiple English words *don't have any money* jointly align to the French words *sont démunis*. Such **phrasal alignments** will be necessary for phrasal MT, but it turns out they can't be directly generated by the IBM Model 1, Model 3, or HMM word alignment algorithms.

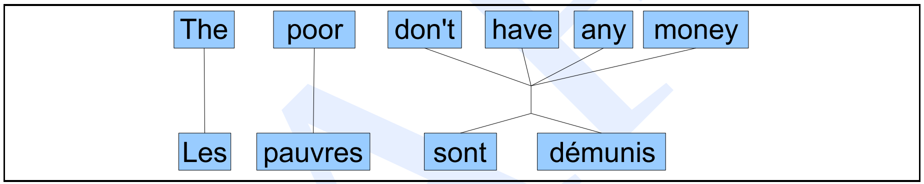


Figure 25.22 An alignment between an English and a French sentence, in which there is a many-to-many alignment between English and French words. Adapted from Brown et al. (1993).

25.5.1 IBM Model 1

We'll describe two alignment models in this section: IBM Model 1 and the HMM model (we'll also sketch the fertility-based IBM Model 3 in the advanced section). Both are **statistical alignment** algorithms. For phrase-based statistical MT, we use the alignment algorithms just to find the best alignment for a sentence pair (F,E), in order to help extract a set of phrases. But it is also possible to use these word alignment algorithms as a translation model $P(F,E)$ as well. As we will see, the relationship between alignment and translation can be expressed as follows:

$$P(F|E) = \sum_A P(F,A|E)$$

We'll start with IBM Model 1, so-called because it is the first and simplest of five models proposed by IBM researchers in a seminal paper (Brown et al., 1993).

Here's the general IBM Model 1 generative story for how we generate a Spanish sentence from an English sentence $E = e_1, e_2, ..., e_I$ of length I:

1. Choose a length J for the Spanish sentence, henceforth $F = f_1, f_2, ..., f_J$.
2. Now choose an alignment $A = a_1, a_2, ..., a_J$ between the English and Spanish sentences.

3. Now for each position j in the Spanish sentence, chose a Spanish word f_j by translating the English word that is aligned to it.

Fig. 25.23 shows a visualization of this generative process.

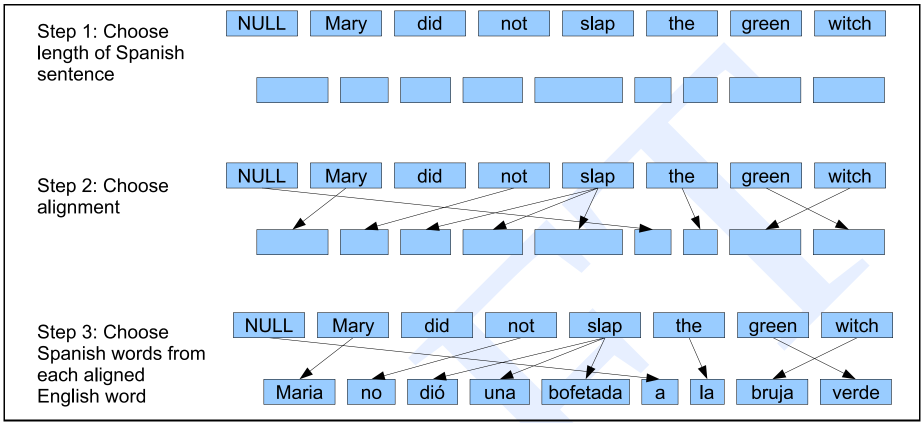


Figure 25.23 The three steps of IBM Model 1 generating a Spanish sentence and alignment from an English sentence.

Let's see how this generative story assigns a probability $P(F|E)$ of generating the Spanish sentence F from the English sentence E. We'll use this terminology:

- e_{a_j} is the English word that is aligned to the Spanish word f_j.
- $t(f_x, e_y)$ is the probability of translating e_y by f_x (i.e. $P(f_x|e_y)$)

We'll work our way backwards from step 3. So suppose we already knew the length J and the alignment A, as well as the English source E. The probability of the Spanish sentence would be:

$$P(F|E,A) = \prod_{j=1}^{J} t(f_j|e_{a_j}) \tag{25.17}$$

Now let's formalize steps 1 and 2 of the generative story. This is the probability $P(A|E)$ of an alignment A (of length J) given the English sentence E. IBM Model 1 makes the (very) simplifying assumption that each alignment is equally likely. How many possible alignments are there between an English sentence of length I and a Spanish sentence of length J? Again assuming that each Spanish word must come from one of the I English words (or the 1 NULL word), there are $(I+1)^J$ possible alignments. Model 1 also assumes that the probability of choosing length J is some small constant ϵ. The combined probability of choosing a length J and then choosing any particular one of the $(I+1)^J$ possible alignments is:

$$P(A|E) = \frac{\epsilon}{(I+1)^J} \tag{25.18}$$

We can combine these probabilities as follows:

$$P(F,A|E) = P(F|E,A) \times P(A|E)$$

$$= \frac{\epsilon}{(I+1)^J}\prod_{j=1}^{J} t(f_j|e_{a_j}) \tag{25.19}$$

This probability, $P(F,A|E)$, is the probability of generating a Spanish sentence F via a particular alignment. In order to compute the total probability $P(F|E)$ of generating F , we just sum over all possible alignments:

$$\begin{aligned} P(F|E) &= \sum_A P(F,A|E) \\ &= \sum_A \frac{\epsilon}{(I+1)^J}\prod_{j=1}^{J} t(f_j|e_{a_j}) \end{aligned} \tag{25.20}$$

Eq. 25.20 shows the generative probability model for Model 1, as it assigns a probability to each possible Spanish sentence.

In order to find the best alignment between a pair of sentences F and E, we need a way to **decode** using this probabilistic model. It turns out there is a very simple polynomial algorithm for computing the best (Viterbi) alignment with Model 1, because the best alignment for each word is independent of the decision about best alignments of the surrounding words:

$$\begin{aligned} \hat{A} &= \underset{A}{\operatorname{argmax}} P(F,A|E) \\ &= \underset{A}{\operatorname{argmax}} \frac{\epsilon}{(I+1)^J}\prod_{j=1}^{J} t(f_j|e_{a_j}) \\ &= \underset{a_j}{\operatorname{argmax}} t(f_j|e_{a_j}) \qquad 1 < j < J \end{aligned} \tag{25.21}$$

Training for Model 1 is done by the EM algorithm, which we will cover in Sec. 25.6.

25.5.2 HMM Alignment

Now that we've seen Model 1, it should be clear that it makes some really appalling simplifying assumptions. One of the most egregious is the assumption that all alignments are equally likely. One way in which this is a bad assumption is that alignments tend to preserve **locality**; neighboring words in English are often aligned with neighboring words in Spanish. If we look back at the Spanish/English alignment in Fig. 25.17, for example, we can see that this locality in the neighboring alignments. The HMM alignment model captures this kind of locality by conditioning each alignment decision on previous decisions. Let's see how this works.

The HMM alignment model is based on the familiar HMM model we've now seen in many chapters. As with IBM Model 1, we are trying to compute $P(F,A|E)$. The HMM model is based on a restructuring of this probability using the chain rule as follows:

$$P(f_1^J, a_1^J|e_1^I) = P(J|e_1^I) \times \prod_{j=1}^{J} P(f_j, a_j|f_1^{j-1}, a_1^{j-1}, e_1^I)$$

$$= P(J|e_1^I) \times \prod_{j-1}^{J} P(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I) \times P(f_j|f_1^{j-1}, a_1^j, e_1^I) \quad (25.22)$$

Via this restructuring, we can think of $P(F,A|E)$ as being computable from probabilities of three types: a length probability $P(J|e_1^I)$, an alignment probability $P(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I)$, and a lexicon probability $P(f_j|f_1^{j-1}, a_1^j, e_1^I)$.

We next make some standard Markov simplifying assumptions. We'll assume that the probability of a particular alignment a_j for Spanish word j is only dependent on the previous aligned position a_{j-1}. We'll also assume that the probability of a Spanish word f_j is dependent only on the aligned English word e_{a_j}at position a_j:

$$P(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I) = P(a_j|a_{j-1}, I) \quad (25.23)$$

$$P(f_j|f_1^{j-1}, a_1^j, e_1^I) = P(f_j|e_{a_j}) \quad (25.24)$$

Finally, we'll assume that the length probability can be approximated just as $P(J|I)$. Thus the probabilistic model for HMM alignment is:

$$P(f_1^J, a_1^J|e_1^I) = P(J|I) \times \prod_{j=1}^{J} P(a_j|a_{j-1}, I) P(f_j|e_{a_j}) \quad (25.25)$$

To get the total probability of the Spanish sentence $P(f_1^J|e_1^I)$ we need to sum over all alignments:

$$P(f_1^J|e_1^I) = P(J|I) \times \sum_{A} \prod_{j=1}^{J} P(a_j|a_{j-1}, I) P(f_j|e_{a_j}) \quad (25.26)$$

As we suggested at the beginning of the section, we've conditioned the alignment probability $P(a_j|a_{j-1}, I)$ on the previous aligned word, to capture the locality of alignments. Let's rephrase this probability for a moment as $P(i|i', I)$, where i will stand for the absolute positions in the English sentence of consecutive aligned states in the Spanish sentence. We'd like to make these probabilities dependent not on the absolute word positions i and i', but rather on the **jump width** between words; the jump width is the distance between their positions $i' - i$. This is because our goal is to capture the fact that *'the English words that generate neighboring Spanish words are likely to be nearby'*. We thus don't want to be keeping separate probabilities for each absolute word position like $P(7|6, 15)$ and $P(8|7, 15)$. Instead, we compute alignment probabilities by using a non-negative function of the jump width:

Jump width

$$P(i|i', I) = \frac{c(i - i')}{\sum_{i''=1}^{I} c(i'' - i')} \quad (25.27)$$

Let's see how this HMM model gives the probability of a particular alignment of our English-Spanish sentences; we've simplified the sentence slightly.

Thus the probability $P(F,A|E)$ for this particular alignment of our simplified sentence *Maria dió una bofetada a la bruja verde* is the product of:

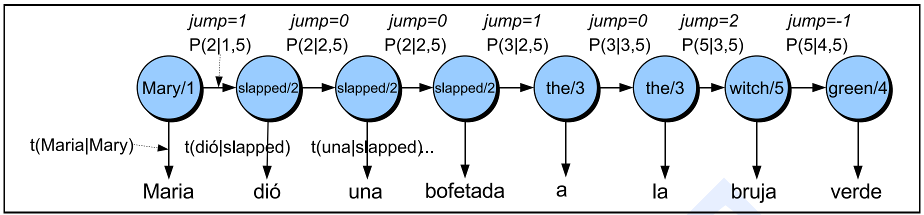


Figure 25.24 The HMM alignment model generating from *Mary slappped the green witch*, showing the alignment and lexicon components of the probability $P(F,A|E)$ for this particular alignment.

$$\begin{aligned} P(F,A|E) &= P(J|I) \times P(Maria|Mary) \times P(2|1,5) \times \\ &\quad t(dió|slapped) \times P(2|2,5) \times T(una|slapped) \times P(2|2,5) \times \ldots \end{aligned} \tag{25.28}$$

There are also more sophisticated augmentations to the basic HMM alignment model. These include adding NULL words in the English source which can be used to align with Spanish words that don't align with English words, or conditioning the alignment on $C(e_{a_{j-1}})$, the word class of the preceding target word: $P(a_j|a_{j-1},I,C(e_{a_{j-1}}))$ (Och and Ney, 2003; Toutanova et al., 2002).

The main advantage of the HMM alignment model is that there are well-understood algorithms both for decoding and for training. For decoding, we can use the Viterbi algorithm introduced in Ch. 5 and Ch. 6 to find the best (Viterbi) alignment for a sentence pair (F,E). For training, we can use the Baum-Welch algorithm, as summarized in the next section.

25.6 Training Alignment Models

Parallel corpus
Bitext
Hansards
Hong Kong Hansards

All statistical translation models are trained using a large **parallel corpus**. A **parallel corpus**, **parallel text**, or **bitext** is a text that is available in two languages. For example, the proceedings of the Canadian parliament are kept in both French and English. Each sentence spoken in parliament is translated, producing a volume with running text in both languages. These volumes are called **Hansards**, after the publisher of the British parliamentary proceedings. Similarly, the **Hong Kong Hansards** corpus contains the proceedings of the Hong Kong SAR Legislative Council in both English and Chinese. Both of these corpora contain tens to hundreds of millions of words. Other parallel corpora have been made available by the United Nations. It is possible to make parallel corpora out of literary translations, but this is less common for MT purposes, partly because it is difficult to acquire the legal rights to fiction, but mainly because, as we saw at the beginning of the chapter, translating fiction is very difficult and translations are not very literal. Thus statistical systems tend to be trained on very literal translations such as Hansards.

Sentence segmentation
Sentence alignment

The first step in training is to segment the corpus into sentences. This task is called **sentence segmentation** or **sentence alignment**. The simplest methods align sentences based purely on their length in words or characters, without looking at the contents of the words in the sentences. The intuition is that if we see a long sentence in roughly the same position in each language of the parallel text, we might suspect these sentences are translations. This intuition can be implemented by a dynamic programming algorithm. More sophisticated algorithms also make use of information about word alignments. Sentence alignment algorithms are run on a parallel corpus before training MT models. Sentences which don't align to anything are thrown out, and the remaining aligned sentences can be used as a training set. See the end of the chapter for pointers to more details on sentence segmentation.

Once we have done sentence alignment, the input to our training algorithm is a corpus consisting of S sentence pairs $\{(F_s,E_s) : s = 1 \ldots S\}$. For each sentence pair (F_s,E_s) the goal is to learn an alignment $A = a_1^J$ and the component probabilities (t for Model 1, and the lexicon and alignment probabilities for the HMM model).

25.6.1 EM for Training Alignment Models

If each sentence pair (F_s,E_s) was already hand-labeled with a perfect alignment, learning the Model 1 or HMM parameters would be trivial. For example, to get a maximum likelihood estimates in Model 1 for the translation probability t(*verde*,*green*), we would just count the number of times *green* is aligned to *verde*, and normalize by the total count of *green*.

But of course we don't know the alignments in advance; all we have are the **probabilities** of each alignment. Recall that Eq˙ 25.19 showed that if we already had good estimates for the Model 1 t parameter, we could use this to compute probabilities $P(F,A|E)$ for alignments. Given $P(F,A|E)$, we can generate the probability of an alignment just by normalizing:

$$P(A|E,F) = \frac{P(A,F|E)}{\sum_A P(A,F|E)}$$

So, if we had a rough estimate of the Model 1 t parameters, we could compute the probability for each alignment. Then instead of estimating the t probabilities from the (unknown) perfect alignment, we would estimate them from each possible alignment, and combine these estimates weighted by the probability of each alignment. For example if there were two possible alignments, one of probability .9 and one of probability .1, we would estimate the t parameters separately from the two alignments and mix these two estimates with weights of .9 and .1.

Thus if we had model 1 parameters already, we could **re-estimate** the parameters, by using the parameters to compute the probability of each possible alignment, and then using the weighted sum of alignments to re-estimate the model 1 parameters. This idea of iteratively improving our estimates of probabilities is a special case of the **EM algorithm** that we introduced in Ch. 6, and that we saw again for speech recognition in Ch. 9. Recall that we use the EM algorithm when we have a variable that we can't optimize directly because it is **hidden**. In this case the hidden variable is the alignment. But we can use the EM algorithm to estimate the parameters, compute alignments from these estimates, use the alignments to re-estimate the parameters, and so on!

Let's walk through an example inspired by Knight (1999b), using a simplified version of Model 1, in which we ignore the NULL word, and we only consider a subset of the alignments (ignoring alignments for which an English word aligns with no Spanish word). Hence we compute the simplified probability $P(A,F|E)$ as follows:

$$P(A,F|E) = \prod_{j=1}^{J} t(f_j|e_{a_j}) \tag{25.29}$$

The goal of this example is just to give an intuition of EM applied to this task; the actual details of Model 1 training would be somewhat different.

The intuition of EM training is that in the E-step, we compute **expected counts** for the t parameter based on summing over the hidden variable (the alignment), while in the M-step, we compute the maximum likelihood estimate of the t probability from these counts.

Let's see a few stages of EM training of this parameter on a corpus of two sentences:

```
green house        the house
casa  verde        la  casa
```

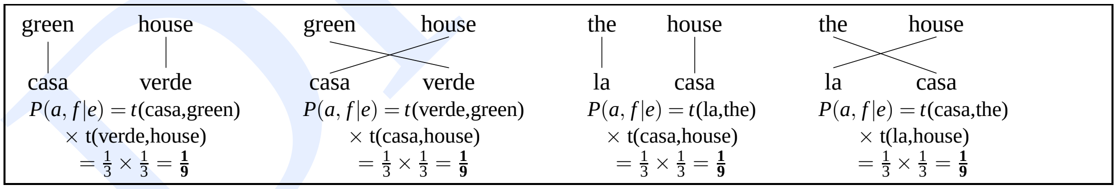
The vocabularies for the two languages are $E = \{\text{green,house,the}\}$ and $S = \{\text{casa,la,verde}\}$. We'll start with uniform probabilities:

t(casa|green) $= \frac{1}{3}$	t(verde|green) $= \frac{1}{3}$	t(la|green) $= \frac{1}{3}$
t(casa|house) $= \frac{1}{3}$	t(verde|house) $= \frac{1}{3}$	t(la|house) $= \frac{1}{3}$
t(casa|the) $= \frac{1}{3}$	t(verde|the) $= \frac{1}{3}$	t(la|the) $= \frac{1}{3}$

Now let's walk through the steps of EM:

E-step 1: Compute the expected counts $E[\text{count}(t(f,e))]$ for all word pairs (f_j, e_{a_j})

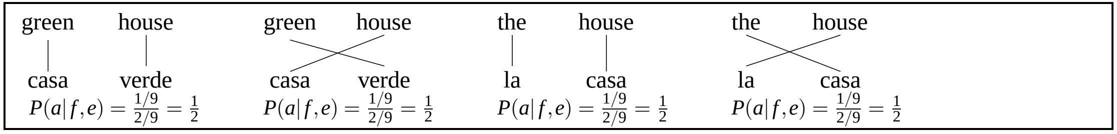
E-step 1a: We first need to compute $P(a,f|e)$, by multiplying all the t probabilities, following Eq. 25.29


green house
casa verde
$P(a,f|e) = t(\text{casa,green}) \times \text{t(verde,house)} = \frac{1}{3} \times \frac{1}{3} = \mathbf{\frac{1}{9}}$
green house
casa verde
$P(a,f|e) = t(\text{verde,green}) \times \text{t(casa,house)} = \frac{1}{3} \times \frac{1}{3} = \mathbf{\frac{1}{9}}$
the house
la casa
$P(a,f|e) = t(\text{la,the}) \times \text{t(casa,house)} = \frac{1}{3} \times \frac{1}{3} = \mathbf{\frac{1}{9}}$
the house
la casa
$P(a,f|e) = t(\text{casa,the}) \times \text{t(la,house)} = \frac{1}{3} \times \frac{1}{3} = \mathbf{\frac{1}{9}}$

E-step 1b: Normalize $P(a,f|e)$ to get $P(a|e,f)$, using the following:

$$P(a|e,f) = \frac{P(a,f|e)}{\sum_a P(a,f|e)}$$

The resulting values of $P(a|f,e)$ for each alignment are as follows:


green house
casa verde
$P(a|f,e) = \frac{1/9}{2/9} = \frac{1}{2}$
green house
casa verde
$P(a|f,e) = \frac{1/9}{2/9} = \frac{1}{2}$
the house
la casa
$P(a|f,e) = \frac{1/9}{2/9} = \frac{1}{2}$
the house
la casa
$P(a|f,e) = \frac{1/9}{2/9} = \frac{1}{2}$

E-step 1c: Compute expected (fractional) counts, by weighting each count by $P(a|e,f)$

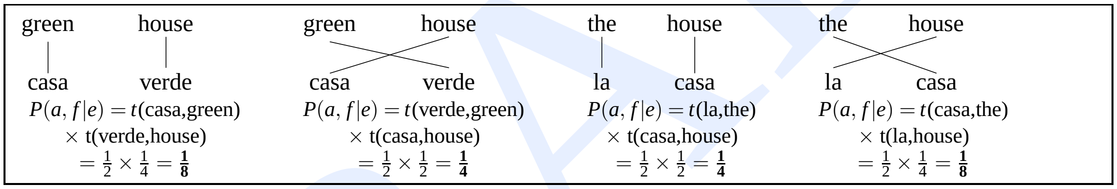
tcount(casa$\|$green) $= \frac{1}{2}$	tcount(verde$\|$green) $= \frac{1}{2}$	tcount(la$\|$green) $= 0$	total(green) $= 1$
tcount(casa$\|$house) $= \frac{1}{2}+\frac{1}{2}$	tcount(verde$\|$house) $= \frac{1}{2}$	tcount(la$\|$house) $= \frac{1}{2}$	total(house) $= 2$
tcount(casa$\|$the) $= \frac{1}{2}$	tcount(verde$\|$the) $= 0$	tcount(la$\|$the) $= \frac{1}{2}$	total(the) $= 1$

M-step 1: Compute the MLE probability parameters by normalizing the tcounts to sum to one.

t(casa$\|$green) $= \frac{1/2}{1} = \frac{1}{2}$	t(verde$\|$green) $= \frac{1/2}{1} = \frac{1}{2}$	t(la$\|$green) $= \frac{0}{1} = 0$
t(casa$\|$house) $= \frac{1}{2} = \frac{1}{2}$	t(verde$\|$house) $= \frac{1/2}{2} = \frac{1}{4}$	t(la$\|$house) $= \frac{1/2}{2} = \frac{1}{4}$
t(casa$\|$the) $= \frac{1/2}{1} = \frac{1}{2}$	t(verde$\|$the) $= \frac{0}{1} = 0$	t(la$\|$the) $= \frac{1/2}{1} = \frac{1}{2}$

Note that each of the correct translations have increased in probability from the initial assignment; for example the translation *casa* for *house* has increased in probability from $\frac{1}{3}$ to $\frac{1}{2}$.

E-step 2a: We re-compute $P(a,f|e)$, again by multiplying all the t probabilities, following Eq. 25.29


green
house
casa
verde
$P(a,f|e) = t(\text{casa,green})$
$\times$ t(verde,house)
$= \frac{1}{2} \times \frac{1}{4} = \mathbf{\frac{1}{8}}$
green
house
casa
verde
$P(a,f|e) = t(\text{verde,green})$
$\times$ t(casa,house)
$= \frac{1}{2} \times \frac{1}{2} = \mathbf{\frac{1}{4}}$
the
house
la
casa
$P(a,f|e) = t(\text{la,the})$
$\times$ t(casa,house)
$= \frac{1}{2} \times \frac{1}{2} = \mathbf{\frac{1}{4}}$
the
house
la
casa
$P(a,f|e) = t(\text{casa,the})$
$\times$ t(la,house)
$= \frac{1}{2} \times \frac{1}{4} = \mathbf{\frac{1}{8}}$

Note that the two correct alignments are now higher in probability than the two incorrect alignments. Performing the second and further round of E-steps and M-steps is left as Exercise 6 for the reader.

We have shown that EM can be used to learn the parameters for a simplified version of Model 1. Our intuitive algorithm, however, requires that we enumerate all possible alignments. For a long sentence, enumerating every possible alignment would be very inefficient. Luckily in practice there is a very efficient version of EM for Model 1 that efficiently and implicitly sums over all alignments.

We also use EM, in the form of the Baum-Welch algorithm, for learning the parameters of the HMM model.

25.7 Symmetrizing Alignments for Phrase-based MT

The reason why we needed Model 1 or HMM alignments was to build word alignments on the training set, so that we could extract aligned pairs of phrases.

Unfortunately, HMM (or Model 1) alignments are insufficient for extracting pairings of Spanish phrases with English phrases. This is because in the HMM model, each Spanish word must be generated from a single English word; we cannot generate a Spanish phrase from multiple English words. The HMM model thus cannot align a multiword phrase in the source language with a multiword phrase in the target language.

Symmetrizing

We can, however, extend the HMM model to produce phrase-to-phrase alignments for a pair of sentences (F, E), via a method that's often called **symmetrizing**. First, we train two separate HMM aligners, an English-to-Spanish aligner and a Spanish-to-English aligner. We then align (F,E) using both aligners. We can then combine these alignments in clever ways to get an alignment that maps phrases to phrases.

Intersection

To combine the alignments, we start by taking the **intersection** of the two alignments, as shown in Fig. 25.25. The intersection will contain only places where the two alignments agree, hence the high-precision aligned words. We can also separately compute the **union** of these two alignments. The union will have lots of less accurately aligned words. We can then build a classifier to select words from the union, which we incrementally add back in to this minimal intersective alignment.

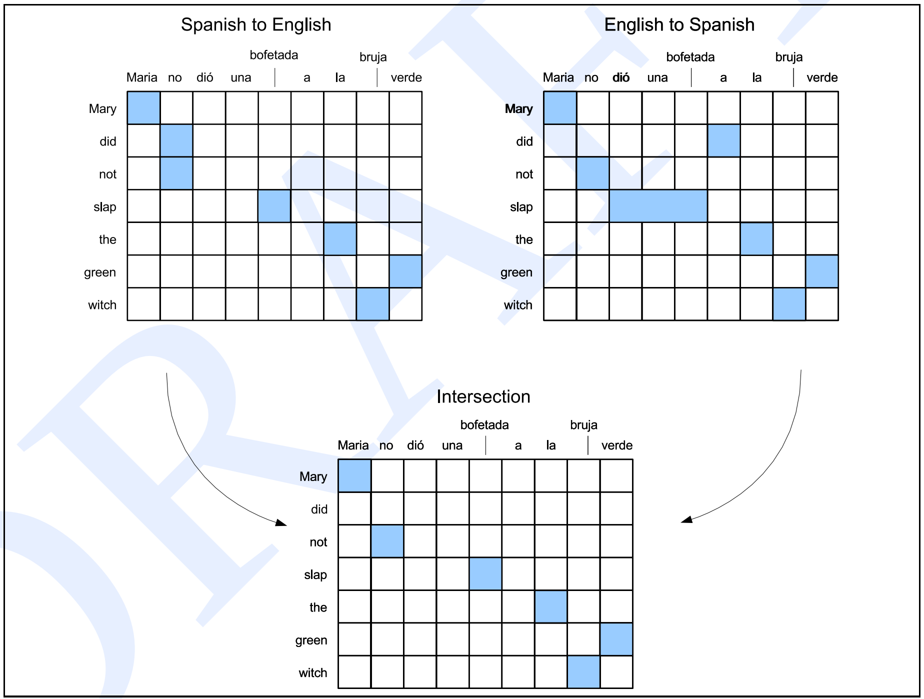
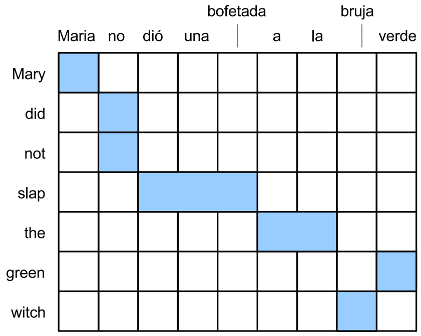


Figure 25.25 Intersection of English-to-Spanish and Spanish-to-English alignments to produce a high-precision alignment. Alignment can then be expanded with points from both alignments to produce an alignment like that shown in Fig. 25.26. After Koehn (2003b).

Fig. 25.26 shows an example of the resulting word alignment. Note that it does allow many-to-one alignments in both directions. We can now harvest all phrase pairs that are consistent with this word alignment. A consistent phrase pair is one in which all the words are aligned only with each other, and not to any external words. Fig. 25.26 also shows some phrases consistent with the alignment.

Once we collect all the aligned phrases pairs from the entire training corpus, we can compute the maximum likelihood estimate for the phrase translation probability of



(Maria, Mary), (no, did not),
(slap, dió una bofetada), (verde, green),
(a la, the), (bruja, witch),
(Maria no, Mary did not),
(no dió una bofetada, did not slap),
(dió una bofetada a la, slap the),
(bruja verde, green witch),
(a la bruja verde, the green witch),. . .

Figure 25.26 A better phrasal alignment for the *green witch* sentence, computed by starting with the intersection alignment in Fig. 25.25 and adding points from the union alignment, using the algorithm of Och and Ney (2003). On the right, some of the phrases consistent with this alignment, after Koehn (2003b).

a particular pair as follows:

$$\phi(\bar{f},\bar{e}) = \frac{\text{count}(\bar{f},\bar{e})}{\sum_{\bar{f}}\text{count}(f,\bar{e})} \tag{25.30}$$

Phrase translation table

We can now store each phrase $(\bar{f},\bar{e})$, together with its probability $\phi(\bar{f},\bar{e})$, in a large **phrase translation table**. The decoding algorithm discussed in the next section can use this phrase translation table to compute the translation probability.

25.8 Decoding for Phrase-Based Statistical MT

The remaining component of a statistical MT system is the decoder. Recall that the job of the decoder is to take a foreign (Spanish) source sentence F and produce the best (English) translation E according to the product of the translation and language models:

$$\hat{E} = \underset{E\in\text{English}}{\text{argmax}} \; \overbrace{P(F|E)}^{\text{translation model}} \; \overbrace{P(E)}^{\text{language model}} \tag{25.31}$$

Finding the sentence which maximizes the translation and language model probabilities is a **search** problem, and decoding is thus a kind of search. Decoders in MT are based on **best-first search**, a kind of **heuristic** or **informed search**; these are search algorithms that are informed by knowledge from the problem domain. Best-first search algorithms select a node n in the search space to explore based on an evaluation function $f(n)$. MT decoders are variants of a specific kind of best-first search called **A*** search. A* search was first implemented for machine translation by IBM (Brown et al., 1995), based on IBM's earlier work on A* search for speech recognition (Jelinek, 1969). As we discussed in Sec. 10.2, for historical reasons A* search and its variants are com-

Stack decoding

monly called **stack decoding** in speech recognition and sometimes also in machine translation.

Let's begin in Fig. 25.27 with a generic version of stack decoding for machine translation. The basic intuition is to maintain a **priority queue** (traditionally referred to as a **stack**) with all the partial translation hypotheses, together with their scores.

```
function STACK DECODING(source sentence) returns target sentence

initialize stack with a null hypothesis
loop do
   pop best hypothesis $h$ off of stack
   if $h$ is a complete sentence, return $h$
   for each possible expansion $h'$ of $h$
         assign a score to $h'$
         push $h'$ onto stack
```

Figure 25.27 Generic version of stack or A* decoding for machine translation. A hypothesis is expanded by choosing a single word or phrase to translate. We'll see a more fleshed-out version of the algorithm in Fig. 25.30.

Let's now describe stack decoding in more detail. While the original IBM statistical decoding algorithms were for word-based statistical MT, we will describe the application to phrase-based decoding in the publicly available MT decoder **Pharaoh** (Koehn, 2004).

In order to limit the search space in decoding, we don't want to search through the space of all English sentences; we only want to consider the ones that are possible translations for F. To help reduce the search space, we only want to consider sentences that include words or phrases which are possible translations of words or phrases in the Spanish sentence F. We do this by searching the **phrase translation table** described in the previous section, for all possible English translations for all possible phrases in F.

A sample lattice of possible translation options is shown in Fig. 25.28 drawn from Koehn (2003a, 2004). Each of these options consists of a Spanish word or phrase, the English translation, and the phrase translation probability ϕ. We'll need to search through combinations of these to find the best translation string.

Now let's walk informally through the stack decoding example in Fig. 25.29, producing an English translation of *Mary dió una bofetada a la bruja verde* left to right. For the moment we'll make the simplifying assumption that there is a single stack, and that there is no pruning.

We start with the null hypothesis as the initial **search state**, in which we have selected no Spanish words and produced no English translation words. We now **expand** this hypothesis by choosing each possible source word or phrase which could generate an English sentence-initial phrase. Fig. 25.29a shows this first ply of the search. For example the top state represents the hypothesis that the English sentence starts with *Mary*, and the Spanish word *Maria* has been covered (the asterisk for the first word is marked with an M). Each state is also associated with a cost, discussed below. Another

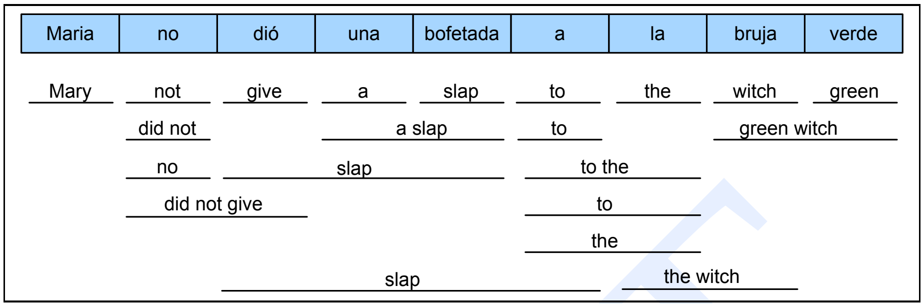


Figure 25.28 The lattice of possible English translations for words and phrases in a particular sentence *F*, taken from the entire aligned training set. After Koehn (2003a)

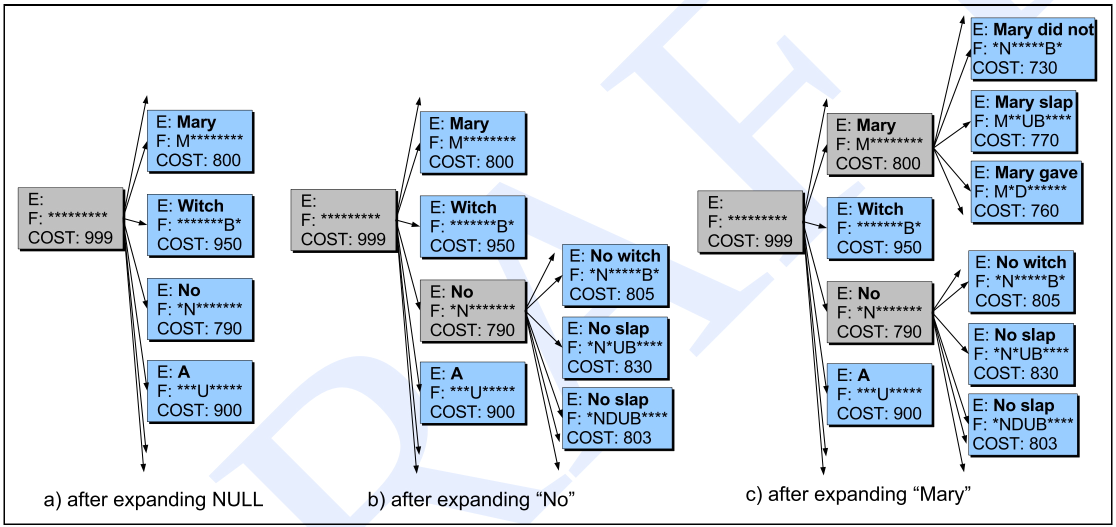


Figure 25.29 Three stages in stack decoding of *Maria no dió una bofetada a la bruja verde* (simplified by assuming a single stack and no pruning). The nodes in blue, on the fringe of the search space, are all on the stack, and are **open** nodes still involved in the search. Nodes in gray are **closed** nodes which have been popped off the stack.

state at this ply represents the hypothesis that the English translation starts with the word *No*, and that Spanish *no* has been covered. This turns out to be the lowest-cost node on the queue, so we pop it off the queue and push all its expansions back on the queue. Now the state *Mary* is the lowest cost, so we expand it; *Mary did not* is now the lowest cost translation so far, so will be the next to be expanded. We can then continue to expand the search space until we have states (hypotheses) that cover the entire Spanish sentence, and we can just read off an English translation from this state.

We mentioned that each state is associated with a cost which, as we'll see below, is used to guide the search, The cost combines the **current cost** with an estimate of the **future cost**. The **current cost** is the total probability of the phrases that have been translated so far in the hypothesis, i.e. the product of the translation, distortion, and language model probabilities. For the set of partially translated phrases $S = (F,E)$, this probability would be:

$$\text{cost}(E,F) = \prod_{i \in S} \phi(\bar{f_i}, \bar{e_i}) d(a_i - b_{i-1}) P(E) \tag{25.32}$$

The **future cost** is our estimate of the cost of translating the *remaining* words in the Spanish sentence. By combining these two factors, the state cost gives an estimate of the total probability of the search path for the eventual complete translation sentence E passing through the current node. A search algorithm based only on the current cost would tend to select translations that had a few high-probability words at the beginning, at the expense of translations with a higher overall probability. [4] For the future cost, it turns out to be far too expensive to compute the true minimum probability for all possible translations. Instead, we approximate this cost by ignoring the distortion cost and just finding the sequence of English phrases which has the minimum product of the language model and translation model costs, which can be easily computed by the Viterbi algorithm.

This sketch of the decoding process suggests that we search the entire state space of possible English translations. But we can't possibly afford to expand the entire search space, because there are far too many states; unlike in speech recognition, the need for distortion in MT means there is (at least) a distinct hypothesis for every possible ordering of the English words![5]

Beam-search pruning

For this reason MT decoders, like decoders for speech recognition, all require some sort of pruning. Pharaoh and similar decoders use a version of **beam-search pruning**, just as we saw in decoding for speech recognition and probabilistic parsing. Recall that in beam-search pruning, at every iteration we keep only the most promising states, and prune away unlikely (high-cost) states (those 'outside the search beam'). We could modify the search sequence depicted in Fig. 25.29, by pruning away all bad (high-cost) states at every ply of the search, and expanding only the best state. In fact, in Pharaoh, instead of expanding only the best state, we expand all states within the beam; thus Pharaoh is technically **beam search** rather than **best-first search** or A^* search.

More formally, at each ply of the search we keep around a stack (priority queue) of states. The stack only fits n entries. At every ply of the search, we expand all the states on the stack, push them onto the stack, order them by cost, keep the best n entries and delete the rest.

We'll need one final modification. While in speech we just used one stack for stack decoding, in MT we'll use multiple stacks, because we can't easily compare the cost of hypotheses that translate different numbers of foreign words. So we'll use m stacks, where stack s_m includes all hypotheses that cover m foreign words. When we expand a hypothesis by choosing a phrase to translate, we'll insert the new state into the correct stack for the number of foreign words covered. Then we'll use beam-search inside each of these stacks, keep only n hypotheses for each of the m stacks. The final multi-stack version of beam search stack decoding is shown in Fig. 25.30.

[4] We saw this same kind of cost function for A^* search in speech recognition, where we used the A^* evaluation function: $f^*(p) = g(p) + h^*(p)$.

[5] Indeed, as Knight (1999a) shows, decoding even in IBM Model 1 with a bigram language model is equivalent to the difficult class of problems known as **NP-complete**.

```
function BEAM SEARCH STACK DECODER(source sentence) returns target sentence

initialize hypothesisStack[0..nf]
push initial null hypothesis on hypothesisStack[0]
for $i \leftarrow 0$ to nf-1
    for each hyp in hypothesisStack[i]
        for each new_hyp that can be derived from hyp
            nf[new_hyp] ← number of foreign words covered by new_hyp
            add new_hyp to hypothesisStack[nf[new_hyp]]
            prune hypothesisStack[nf[new_hyp]]
find best hypothesis best_hyp in hypothesisStack[nf]
return best path that leads to best_hyp via backtrace
```

Figure 25.30 Pharaoh beam search multi-stack decoding algorithm, adapted from (Koehn, 2003a). For efficiency, most decoders don't store the entire foreign and English sentence in each state, requiring that we backtrace to find the state path from the initial to the final state so we can generate the entire English target sentence.

Recombining hypotheses

There are a number of additional issues in decoding that must be dealt with. All decoders attempt to limit somewhat the exponential explosion in the search space by **recombining hypotheses**. . We saw hypothesis recombination in the **Exact *N*-Best** algorithm of Sec. 10.1. In MT, we can merge any two hypotheses that are sufficiently similar (cover the same foreign words, have the same last-two English words, and have the same end of the last foreign phrase covered).

In addition, it turns out that decoders for phrasal MT optimize a slightly different function than the one we presented in Eq. 25.31. In practice, it turns out that we need to add another factor, which serves to penalize sentences which are too short. Thus the decoder is actually choosing the sentence which maximizes:

$$\hat{E} = \operatorname*{argmax}_{E\in\text{English}} \overbrace{P(F|E)}^{\text{translation model}} \overbrace{P(E)}^{\text{language model}} \overbrace{\omega^{\text{length}(E)}}^{\text{short sentence penalty}} \tag{25.33}$$

This final equation is extremely similar to the use of the word insertion penalty in speech recognition in Eq. 9.49.

25.9 MT Evaluation

Evaluating the quality of a translation is an extremely subjective task, and disagreements about methodology are rampant. Nevertheless, evaluation is essential, and research on evaluation methodology has played an important role from the earliest days of MT (Miller and Beebe-Center, 1958). Broadly speaking, translations are evaluted along two dimensions, corresponding to the **fidelity** and **fluency** discussed in Sec. 25.3.

25.9.1 Using Human Raters

The most accurate evaluations use human raters to evaluate each translation along each dimension. For example, along the dimension of **fluency**, we can ask how intelligible, how clear, how readable, or how natural is the MT output (the target translated text). There are two broad ways to use human raters to answer these questions. One method is to give the raters a scale, for example from 1 (totally unintelligible) to 5 (totally intelligible), and ask them to rate each sentence or paragraph of the MT output. We can use distinct scales for any of the aspects of fluency, such as **clarity**, **naturalness**, or **style**. The second class of methods relies less on the conscious decisions of the participants. For example, we can measure the time it takes for the raters to read each output sentence or paragraph. Clearer or more fluent sentences should be faster or easier to read. We can also measure fluency with the **cloze** task (Taylor, 1953, 1957). The cloze task is a metric used often in psychological studies of reading. The rater sees an output sentence with a word replaced by a space (for example, every 8th word might be deleted). Raters have to guess the identity of the missing word. Accuracy at the cloze task, i.e. average success of raters at guessing the missing words, generally correlates with how intelligible or natural the MT output is.

Cloze

A similar variety of metrics can be used to judge the second dimension, **fidelity**. Two common aspects of fidelity which are measured are **adequacy** and **informativeness**. The **adequacy** of a translation is whether it contains the information that existed in the original. We measure adequacy by using raters to assign scores on a scale. If we have bilingual raters, we can give them the source sentence and a proposed target sentence, and rate, perhaps on a 5-point scale, how much of the information in the source was preserved in the target. If we only have monolingual raters, but we have a good human translation of the source text, we can give the monolingual raters the human reference translation and a target machine translation, and again rate how much information is preserved. The **informativeness** of a translation is a task-based evaluation of whether there is sufficient information in the MT output to perform some task. For example we can give raters multiple-choice questions about the content of the material in the source sentence or text. The raters answer these questions based only on the MT output. The percentage of correct answers is an informativeness score.

Adequacy

Informativeness

Another set of metrics attempt to judge the overall quality of a translation, combining fluency and fidelity. For example, the typical evaluation metric for MT output to be post-edited is the **edit cost** of **post-editing** the MT output into a good translation. For example, we can measure the number of words, the amount of time, or the number of keystrokes required for a human to correct the output to an acceptable level.

Edit cost

Post-editing

25.9.2 Automatic Evaluation: Bleu

While humans produce the best evaluations of machine translation output, running a human evaluation can be very time-consuming, taking days or even weeks. It is useful to have an automatic metric that can be run relatively frequently to quickly evaluate potential system improvements. In order to have such convenience, we would be willing for the metric to be much worse than human evaluation, as long as there was some correlation with human judgments.

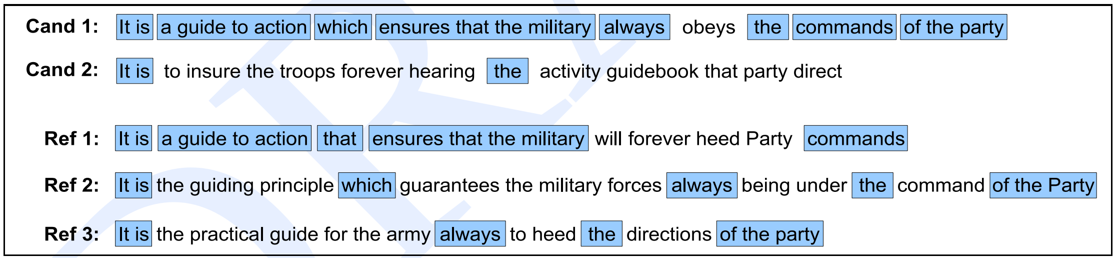
In fact there are a number of such heuristic methods, such as **Bleu**, **NIST**, **TER**, **Precision and Recall**, and **METEOR** (see references at the end of the chapter). The intuition of these automatic metrics derives from Miller and Beebe-Center (1958), who pointed out that a good MT output is one which is very similar to a human translation.

In the field of automatic speech recognition, we define 'very similar' via the word error rate, which is the minimum edit distance to a human transcript. But in translation, we don't rely on a single human translation, because a source sentence could be legitimately translated in many ways. A very good MT output might look like one human translation, but very unlike another one. For this reason, MT metrics generally require that we have multiple human translations of each sentence in a test set. This may seem time-consuming, but the hope is that we can reuse this translated test set over and over again to evaluate new ideas.

Now given an MT output sentence in a test set, we compute the translation closeness between the MT output and the human sentences. An MT output is ranked as better if on average it is closer to the human translations. The metrics differ on what counts as 'translation closeness'.

For the rest of this section, let's walk through one of these metrics, the **Bleu** metric, following closely the original presentation in Papineni et al. (2002). In Bleu we rank each MT output by a weighted average of the number of *N*-gram overlaps with the human translations.

Fig. 25.31 shows an intuition, from two candidate translations of a Chinese source sentence (Papineni et al., 2002), shown with three reference human translations of the source sentence. Note that Candidate 1 shares many more words (shown in blue) with the reference translations than Candidate 2.



Cand 1: It is a guide to action which ensures that the military always obeys the commands of the party

Cand 2: It is to insure the troops forever hearing the activity guidebook that party direct

Ref 1: It is a guide to action that ensures that the military will forever heed Party commands

Ref 2: It is the guiding principle which guarantees the military forces always being under the command of the Party

Ref 3: It is the practical guide for the army always to heed the directions of the party

Figure 25.31 Intuition for Bleu: one of two candidate translations of a Chinese source sentence shares more words with the reference human translations.

Let's look at how the Bleu score is computed, starting with just unigrams. Bleu is based on precision. A basic unigram precision metric would be to count the number of words in the candidate translation (MT output) that occur in some reference translation, and divide by the total number of words in the candidate translation. If a candidate translation had 10 words, and 6 of them occurred in at least one of the reference translations, we would have a precision of $6/10 = 0.6$. Alas, there is a flaw in using simple precision: it rewards candidates that have extra repeated words. Fig. 25.32 shows an example of a pathological candidate sentence composed of multiple instances of the single word *the*. Since each of the 7 (identical) words in the candidate occur in one of the reference translations, the unigram precision would be 7/7!

Modified N-gram precision

In order to avoid this problem, Bleu uses a **modified N-gram precision** metric. We

Candidate:	the	the	the	the	the	the	the
Reference 1:	the	cat	is	on	the	mat	
Reference 2:	there	is	a	cat	on	the	mat

Figure 25.32 A pathological example showing why Bleu uses a modified precision metric. Unigram precision would be unreasonably high (7/7). Modified unigram precision is appropriately low (2/7).

first count the maximum number of times a word is used in any single reference translation. The count of each *candidate* word is then clipped by this maximum *reference* count. Thus the modified unigram precision in the example in Fig. 25.32 would be 2/7, since Reference 1 has a maximum of 2 *the*s. Going back to Chinese example in Fig. 25.32, Candidate 1 has a modified unigram precision of 17/18, while Candidate 2 has one of 8/14.

We compute the modified precision similarly for higher order N-grams as well. The modified bigram precision for Candidate 1 is 10/17, and for Candidate 2 is 1/13. The reader should check these numbers for themselves on Fig. 25.31.

To compute a score over the whole testset, Bleu first computes the N-gram matches for each sentence, and add together the clipped counts over all the candidates sentences, and divide by the total number of candidate N-grams in the testset. The modified precision score is thus:

$$p_n = \frac{\sum\limits_{C \in \{Candidates\}} \sum\limits_{n\text{-}gram \in C} \text{Count}_{\text{clip}}(n\text{-}gram)}{\sum\limits_{C' \in \{Candidates\}} \sum\limits_{n\text{-}gram' \in C'} \text{Count}(n\text{-}gram')} \tag{25.34}$$

Bleu uses unigram, bigrams, trigrams, and often quadrigrams; it combines these modified N-gram precisions together by taking their geometric mean.

In addition, Bleu adds a further penalty to penalize candidate translations that are too short. Consider the candidate translation *of the*, compared with References 1-3 in Fig. 25.31 above. Because this candidate is so short, and all its words appear in some translation, its modified unigram precision is inflated to 2/2. Normally we deal with these problems by combining precision with *recall*. But as we discussed above, we can't use recall over multiple human translations, since recall would require (incorrectly) that a good translation must contain contains lots of N-grams from *every* translation. Instead, Bleu includes a brevity penalty over the whole corpus. Let c be the total length of the candidate translation corpus. We compute the **effective reference length** r for that corpus by summing, for each candidate sentence, the lengths of the best matches. The brevity penalty is then an exponential in r/c. In summary:

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$

$$\text{Bleu} = \text{BP} \times \exp\left(\frac{1}{N}\sum_{n=1}^{N}\log p_n\right) \tag{25.35}$$

While automatic metrics like Bleu (or NIST, METEOR, etc) have been very useful in quickly evaluating potential system improvements, and match human judgments in many cases, they have certain limitations that are important to consider. First, many of them focus on very local information. Consider slightly moving a phrase in Fig. 25.31 slightly to produce a candidate like: *Ensures that the military it is a guide to action which always obeys the commands of the party*. This sentence would have an identical Bleu score to Candidate 1, although a human rater would give it a lower score.

Furthermore, the automatic metrics probably do poorly at comparing systems that have radically different architectures. Thus Bleu, for example, is known to perform poorly (i.e. not agree with human judgments of translation quality) when evaluating the output of commercial systems like Systran against N-gram-based statistical systems, or even when evaluating human-aided translation against machine translation (Callison-Burch et al., 2006). We can conclude that automatic metrics are most appropriate when evaluating incremental changes to a single system, or comparing systems with very similar architectures.

25.10 Advanced: Syntactic Models for MT

The earliest statistical MT systems (like IBM Models 1, 2 and 3) were based on words as the elementary units. The phrase-based systems that we described in earlier sections improved on these word-based systems by using larger units, thus capturing larger contexts and providing a more natural unit for representing language divergences.

Recent work in MT has focused on ways to move even further up the Vauquois hierarchy, from simple phrases to larger and hierarchical syntactic structures.

Transduction grammar

It turns out that it doesn't work just to constrain each phrase to match the syntactic boundaries assigned by traditional parsers (Yamada and Knight, 2001). Instead, modern approaches attempt to assign a parallel syntactic tree structure to a pair of sentences in different languages, with the goal of translating the sentences by applying reordering operations on the trees. The mathematical model for these parallel structures is known as a **transduction grammar**. These transduction grammars can be viewed as an explicit implementation of the **syntactic transfer** systems that we introduced on page 885, but based on a modern statistical foundation.

Synchronous grammar

Inversion transduction grammar

A transduction grammar (also called a **synchronous grammar**) describes a structurally correlated pair of languages. From a generative perspective, we can view a transduction grammar as generating pairs of aligned sentences in two languages. Formally, a transduction grammar is a generalization of the finite-state transducers we saw in Ch. 3. There are a number of transduction grammars and formalisms used for MT, most of which are generalizations of context-free grammars to the two-language situation. Let's consider one of the most widely used such models for MT, the **inversion transduction grammar** (ITG).

In an ITG grammar, each non-terminal generates two separate strings. There are three types of these rules. A lexical rule like the following:

$$N \rightarrow \mathit{witch}/\mathit{bruja}$$

generates the word *witch* on one stream, and *bruja* on the second stream. A nonterminal rule in square brackets like:

$$S \rightarrow [NP\ VP]$$

generates two separate streams, each of *NP VP*. A non-terminal in angle brackets, like

$$\mathit{Nominal} \rightarrow \langle \mathit{Adj}\ N \rangle$$

generates two separate streams, with *different orderings*: *Adj N* in one stream, and *N Adj* in the other stream.

Fig. 25.33 shows a sample grammar with some simple rules. Note that each lexical rule derives distinct English and Spanish word strings, that rules in square brackets ([]) generate two identical non-terminal right-hand sides, and that the one rule in angle brackets (⟨⟩) generates different orderings in Spanish from English.

$$\begin{array}{rcl}
S & \rightarrow & [NP\ VP] \\
NP & \rightarrow & [Det\ Nominal] \quad |\ \mathit{Maria}/\mathit{María} \\
\mathit{Nominal} & \rightarrow & \langle \mathit{Adj}\ \mathit{Noun} \rangle \\
VP & \rightarrow & [V\ PP] \quad |\ [\mathit{Negation}\ VP] \\
\mathit{Negation} & \rightarrow & \mathit{didn't}/\mathit{no} \\
V & \rightarrow & \mathit{slap}/\mathit{dió\ una\ bofetada} \\
PP & \rightarrow & [P\ NP] \\
P & \rightarrow & \epsilon/a \quad |\ \mathit{from}/\mathit{de} \\
\mathit{Det} & \rightarrow & \mathit{the}/\mathit{la} \quad |\ \mathit{the}/\mathit{le} \\
\mathit{Adj} & \rightarrow & \mathit{green}/\mathit{verde} \\
N & \rightarrow & \mathit{witch}/\mathit{bruja}
\end{array}$$

Figure 25.33 A mini Inversion Transduction Grammar grammar for the *green witch* sentence.

Thus an ITG parse tree is a single joint structure which spans over the two observed sentences:

(25.36) (a) [$_S$ [$_{NP}$ Mary] [$_{VP}$ didn't [$_{VP}$ slap [$_{PP}$ [$_{NP}$ the [$_{Nom}$ green witch]]]]]]
(b) [$_S$ [$_{NP}$ María] [$_{VP}$ no [$_{VP}$ dió una bofetada [$_{PP}$ a [$_{NP}$ la [$_{Nom}$ bruja verde]]]]]]

Each non-terminal in the parse derives two strings, one for each language. Thus we could visualize the two sentences in a single parse, where the angle brackets mean that the order of the *Adj N* constituents *green witch* and *bruja verde* are generated in opposite order in the two languages:

[$_S$ [$_{NP}$ Mary/María] [$_{VP}$ didn't/no [$_{VP}$ slap/dió una bofetada [$_{PP}$ ϵ/a [$_{NP}$ the/la ⟨$_{Nom}$ witch/bruja green/verde⟩]]]]]

There are a number of related kinds of synchronous grammars, including synchronous context-free grammars (Chiang, 2005), multitext grammars (Melamed, 2003), lexicalized ITGs (Melamed, 2003; Zhang and Gildea, 2005), and synchronous tree-adjoining and tree-insertion grammars (Shieber and Schabes, 1992; Shieber, 1994b; Nesson et al., 2006). The synchronous CFG system of Chiang (2005), for example, learns hierarchical pairs of rules that capture the fact that Chinese relative clauses appear to the left of their head, while English relative clauses appear to the right of their head:

<① de ②, the ② that ①>

Other models for translation by aligning parallel parse trees including (Wu, 2000; Yamada and Knight, 2001; Eisner, 2003; Melamed, 2003; Galley et al., 2004; Quirk et al., 2005; Wu and Fung, 2005).

25.11 Advanced: IBM Model 3 and fertility-based alignment

The seminal IBM paper that began work on statistical MT proposed five models for MT. We saw IBM's Model 1 in Sec. 25.5.1. Models 3, 4 and 5 all use the important concept of **fertility**. We'll introduce Model 3 in this section; our description here is influenced by Kevin Knight's nice tutorial (Knight, 1999b). Model 3 has a more complex generative model than Model 1. The generative model from an English sentence $E = e_1, e_2, ..., e_I$ has 5 steps:

Fertility

1. For each English word e_i, we choose a **fertility** ϕ_i.[6] The fertility is the number of (zero or more) Spanish words that will be generated from e_i, and is dependent only on e_i.

Spurious word

2. We also need to generate Spanish words from the NULL English word. Recall that we defined these earlier as **spurious words**. Instead of having a fertility for NULL, we'll generate spurious words differently. Every time we generate an English word, we consider (with some probability) generating a spurious word (from NULL).
3. We now know how many Spanish words to generate from each English word. So now for each of these Spanish potential words, generate it by translating its aligned English word. As with Model 1, the translation will be based only on the English word. Spurious Spanish words will be generated by translating the NULL word into Spanish.
4. Move all the non-spurious words into their final positions in the Spanish sentence.
5. Insert the spurious Spanish words in the remaining open positions in the Spanish sentence.

Fig. 25.34 shows a visualization of the Model 3 generative process

n *t* *d* *p1*

Model 3 has more parameters than Model 1. The most important are the **n**, **t**, **d**, and **p1** probabilities. The fertility probability ϕ_i of a word e_i is represented by the

[6] This ϕ is not related to the ϕ that was used in phrase-based translation.

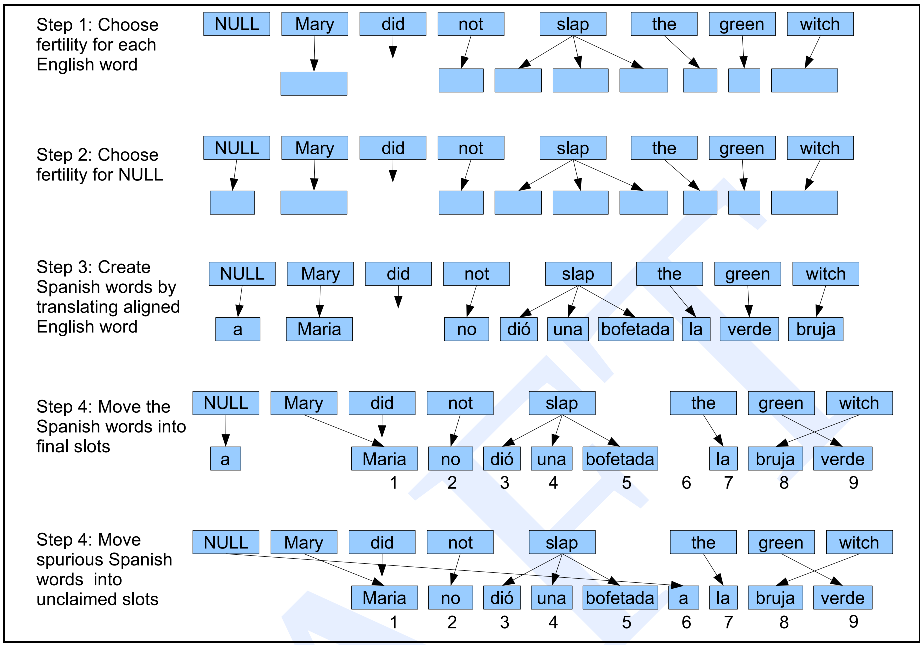

Step 1: Choose fertility for each English word
NULL
Mary
did
not
slap
the
green
witch
Step 2: Choose fertility for NULL
Step 3: Create Spanish words by translating aligned English word
a
Maria
no
dió
una
bofetada
la
verde
bruja
Step 4: Move the Spanish words into final slots
1
2
3
4
5
6
7
8
9
Step 4: Move spurious Spanish words into unclaimed slots

Figure 25.34 The five steps of IBM Model 3 generating a Spanish sentence and alignment from an English sentence.

parameter n. So we will use $n(1|green)$ to represent the probability that English *green* will produce one Spanish word, $n(2|green)$ is the probability that English *green* will produce two Spanish words, $n(0|did)$ is the probability that English *did* will produce no Spanish words, and so on. Like IBM Model 1, Model 3 has a translation probability $t(f_j|e_i)$. Next, the probability that expresses the word position that English words end up in the Spanish sentence is the **distortion** probability, which is conditioned on the English and Spanish sentence lengths. The distortion probability $d(1,3,6,7)$ expresses the probability that the English word e_1 will align to Spanish word f_3, given that the English sentence has length 6, and the Spanish sentence is of length 7.

Distortion

As we suggested above, Model 3 does not use fertility probabilities like $n(1|NULL)$, or $n(3|NULL)$ to decide how many spurious foreign words to generate from English NULL. Instead, each time Model 3 generates a real word, it generates a spurious word for the target sentence with probability p_1. This way, longer source sentences will naturally generate more spurious words. Fig. 25.35 shows a slightly more detailed version of the 5 steps of the Model 3 generative story using these parameters.

Switching for a moment to the task of French to English translation, Fig. 25.36 shows some of the t and ϕ parameters learned for French-English translation from Brown et al. (1993). Note that *the* in general translates to a French article like *le*, but sometimes it has a fertility of 0, indicating that English uses an article where French does not. Conversely, note that *farmers* prefers a fertility of 2, and the most likely translations are *agriculteurs* and *les*, indicating that here French tends to use an article where English does not.

```
1. for each English word $e_i$, $1 < i < I$, we choose a fertility $\phi_i$ with probability $n(\phi_i|e_i)$
2. Using these fertilities and $p_1$, determine $\phi_0$, the number of spurious Spanish words, and hence $m$.
3. for each $i$, $0 < i < I$
    for each $k$, $1 < k < \phi_i$
        Choose a Spanish word $\tau_{ik}$ with probability $t(\tau_{ik}, e_i)$
4. for each $i$, $1 < i < I$
    for each $k$, $1 < k < \phi_i$
        Choose a target Spanish position $\pi_{ik}$ with probability $d(\pi_{ik}, i, I, J)$
5. for each $k$, $1 < k < \phi_0$
        Choose a target Spanish position $\pi_{0k}$ from one of the available Spanish slots, for a total probability of $\frac{1}{\phi_0!}$
```

Figure 25.35 The Model 3 generative story for generating a Spanish sentence from an English sentence. Remember that we are not translating from English to Spanish; this is just the generative component of the noisy channel model. Adapted from Knight (1999b).

	the				*farmers*				*not*		
f	$t(f\|e)$	ϕ	$n(\phi\|e)$	f	$t(f\|e)$	ϕ	$n(\phi\|e)$	f	$t(f\|e)$	ϕ	$n(\phi\|e)$
le	0.497	1	0.746	agriculteurs	0.442	2	0.731	ne	0.497	2	0.735
la	0.207	0	0.254	les	0.418	1	0.228	pas	0.442	0	0.154
les	0.155			cultivateurs	0.046	0	0.039	non	0.029	1	0.107
l’	0.086			producteurs	0.021			rien	0.011		
ce	0.018										
cette	0.011										

Figure 25.36 Examples of Model 3 parameters from the Brown et al. (1993) French-English translation system, for three English words. Note that both *farmers* and *not* are likely to have fertilities of 2.

Now that we have seen the generative story for Model 3, let’s build the equation for the probability assigned by the model. The model needs to assigns a probability $P(F|E)$ of generating the Spanish sentence F from the English sentence E. As we did with Model 1, we’ll start by showing how the model gives the probability $P(F,A|E)$, the probability of generating sentence F via a particular alignment A. Then we’ll sum over all alignments to get the total $P(F|E)$.

In order to compute $P(F,A|E)$, we’ll need to multiply the main three factors n, t, and d, for generating words, translating them into Spanish, and moving them around. So a first pass at $P(F,A|E)$ would be:

$$\prod_{i=1}^{I} n(\phi_i|e_i) \times \prod_{j=1}^{J} t(f_j|e_{a_j}) \times \prod_{j=1}^{J} d(j|a_j, I, J) \tag{25.37}$$

But (25.37) isn’t sufficient as it stands; we need to add factors for generating spurious words, for inserting them into the available slots, and a factor having to do with the number of ways (permutations) a word can align with multiple words. Eq. 25.38 gives the true final equation for IBM Model 3, in Knight’s modification of the original formula. We won’t give the details of these additional factors, but encourage the interested reader to see the original presentation in Brown et al. (1993) and the very clear explanation of the equation in Knight (1999b).

$$
\begin{aligned}
P(F,A|E) \;=\;& \overbrace{\binom{J-\phi_0}{\phi_0} p_0^{J-2\phi_0} p_1^{\phi_0}}^{\text{generate spurious}} \times \overbrace{\frac{1}{\phi_0!}}^{\text{insert spurious}} \times \overbrace{\prod_{i=0}^{I} \phi_i!}^{\text{multi-align permutations}} \\
& \times \prod_{i=1}^{I} n(\phi_i|e_i) \times \prod_{j=1}^{J} t(f_j|e_{a_j}) \times \prod_{j:a_j\neq 0}^{J} d(j|a_j,I,J)
\end{aligned}
\qquad (25.38)
$$

Once again, in order to get the total probability of the Spanish sentence we'll need to sum over all possible alignments:

$$
P(F|E) \;=\; \sum_{A} P(F,A|E)
$$

We can also make it more explicit exactly how we sum over alignments (and also emphasize the incredibly large number of possible alignments) by expressing this formula as follows, where we specify an alignment by specifying the aligned English a_j for each of the J words in the foreign sentence:

$$
P(F|E) \;=\; \sum_{a_1=0}^{J} \sum_{a_2=0}^{J} \cdots \sum_{a_J=0}^{I} P(F,A|E)
$$

25.11.1 Training for Model 3

Given a parallel corpus, training the translation model for IBM Model 3 means setting values for the n, d, t, and p_1 parameters.

As we noted for Model 1 and HMM models, if the training-corpus was hand-labeled with perfect alignments, getting maximum likelihood estimates would be simple. Consider the probability $n(0|did)$ that a word like *did* would have a zero fertility. We could estimate this from an aligned corpus just by counting the number of times *did* aligned to nothing, and normalize by the total count of *did*. We can do similar things for the t translation probabilities. To train the distortion probability $d(1,3,6,7)$, we similarly count the number of times in the corpus that English word e_1 maps to Spanish word f_3 in English sentences of length 6 that are aligned to Spanish sentences of length 7. Let's call this counting function dcount. We'll again need a normalization factor;

$$
d(1,3,6,7) = \frac{\text{dcount}(1,3,6,7)}{\sum_{i=1}^{I} \text{dcount}(i,3,6,7)} \qquad (25.39)
$$

Finally, we need to estimate p_1. Again, we look at all the aligned sentences in the corpus; let's assume that in the Spanish sentences there are a total of N words. From the alignments for each sentence, we determine that a total of S Spanish words are spurious, i.e. aligned to English NULL. Thus $N-S$ of the words in the Spanish sentences were generated by real English words. After S of these $N-S$ Spanish words, we generate a spurious word. The probability p_1 is thus $S/(N-S)$.

Of course, we don't have hand-alignments for Model 3. We'll need to use EM to learn the alignments and the probability model simultaneously. With Model 1 and the HMM model, there were efficient ways to do training without explicitly summing over all alignments. Unfortunately, this is not true for Model 3; we actually would need to compute all possible alignments. For a real pair of sentences, with 20 English words and 20 Spanish words, and allowing NULL and allowing fertilities, there are a very large number of possible alignments (determining the exact number of possible alignments is left as Exercise 7). Instead, we approximate by only considering the best few alignments. In order to find the best alignments without looking at all alignments, we can use an iterative or bootstrapping approach. In the first step, we train the simpler IBM Model 1 or 2 as discussed above. Then we use these Model 2 parameters to evaluate $P(A|E,F)$, giving a way to find the best alignments to bootstrap Model 3. See Brown et al. (1993) and Knight (1999b) for details.

25.12 Advanced: Log-linear Models for MT

While statistical MT was first based on the noisy channel model, much recent work combines the language and translation models via a log-linear model in which we directly search for the sentence with the highest posterior probability:

$$\hat{E} = \underset{E}{\operatorname{argmax}} P(E|F) \tag{25.40}$$

This is done by modeling $P(E|F)$ via a set of M feature functions $h_m(E,F)$, each of which has a parameter λ_m. The translation probability is then:

$$P(E|F) = \frac{\exp[\sum_{m=1}^{M} \lambda_m h_m(E,F)]}{\sum_{E'} \exp[\sum_{m=1}^{M} \lambda_m h_m(E',F)]} \tag{25.41}$$

The best sentence is thus:

$$\begin{aligned}\hat{E} &= \underset{E}{\operatorname{argmax}} P(E|F) \\ &= \underset{E}{\operatorname{argmax}} \exp[\sum_{m=1}^{M} \lambda_m h_m(E,F)] \qquad (25.42)\end{aligned}$$

In practice, the noisy channel model factors (the language model $P(E)$ and translation model $P(F|E)$), are still the most important feature functions in the log-linear model, but the architecture has the advantage of allowing for arbitrary other features as well; a common set of features would include:

- the language model $P(E)$
- the translation model $P(F|E)$

Reverse translation model

- the **reverse translation model** $P(E|F)$,