Dividing and Conquering Long Sentences in a Translation System

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, Robert L. Mercer, and Surya Mohanty *

IBM Thomas J. Watson Research Center
Yorktown Heights, NY 10598

ABSTRACT

The time required for our translation system to handle a sentence of length l is a rapidly growing function of l. We describe here a method for analyzing a sentence into a series of pieces that can be translated sequentially. We show that for sentences with ten or fewer words, it is possible to decrease the translation time by 40% with almost no effect on translation accuracy. We argue that for longer sentences, the effect should be more dramatic.

Introduction

In a recent series of papers, Brown *et al.* introduce a new, statistical approach to machine translation based on the mathematical theory of communication through a noisy channel, and apply it to the problem of translating naturally occurring French sentences into English [1, 2, 3, 4]. They develop a probabilistic model for the noisy channel and show how to estimate the parameters of their model from a large collection of pairs of aligned sentences. By treating a sentence in the source language (French) as a garbled version of the corresponding sentence in the target language (English), they recast the problem of translating a French sentence into English as one of finding that English sentence which is most likely to be present at the input to the noisy channel when the given French sentence is known to be present at its output. For a French sentence of any realistic length, the most probable English translation is one of a set of

English sentences that, although finite, is nonetheless so large as to preclude an exhaustive search. Brown *et al.* employ a suboptimal search based on the stack algorithm used in speech recognition. Even so, as we see in Figure 1, the time required for their system to translate a sentence grows very rapidly with sentence length. As a result, they have focussed their attention on short sentences.
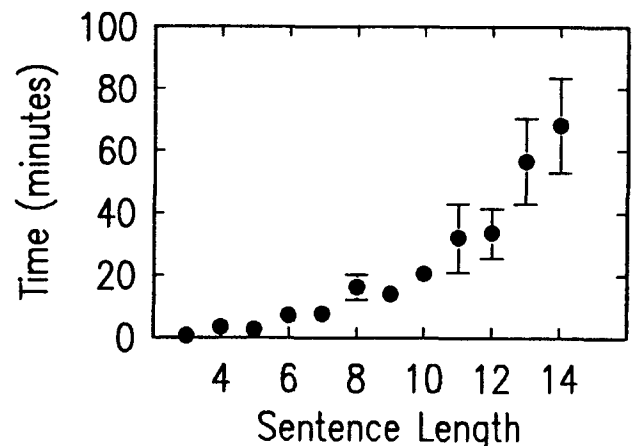

Figure 1: Search time as a function of sentence length.

The designatum of some French words is so specific that they can be reliably translated almost anywhere they occur without regard for the context in which they appear. For example, only the most contrived circumstances could require one to translate the French *technétium* into English as anything but *technetium*. Alas, this charming class of words is woefully small: for the great majority of words, phrases, and even sentences, the more we know of the context

267

in which they appear, the more confidently and eloquently we are able to translate them. But the example provided by simultaneous translators shows that at the expense of eloquence it is possible to produce satisfactory translation segment by segment seriatim.

In this paper, we describe a method for analyzing long sentences into smaller units that can be translated sequentially. Obviously any such analysis risks rupturing some organic whole within the sentence, thereby precipitating an erroneous translation. Thus, phrases like (*pommes frites* | *French fries*), (*pommes de discorde* | *bones of contention*), (*pommes de terre* | *potatoes*), and (*pommes sauvages* | *crab apples*), offer scant hope for subdivision. Even when the analysis avoids splitting a noun from an associated adjective or the opening word of an idiom from its conclusion, we cannot expect that breaking a sentence into pieces will improve translation. The gain that we can expect is in the speed of translation. In general we must weigh this gain in translation speed against the loss in translation accuracy when deciding whether to divide a sentence at a particular point.

Rifts

Brown *et al.* [1] define an *alignment* between an English sentence and its French translation to be a diagram showing for each word in the English sentence those words in the French sentence to which it gives rise (see their Figure 3). The line joining an English word to one of its French dependents in such a diagram is called a *connection*. Given an alignment, we say that the position between two words in a French sentence is a *rift* provided none of the connections to words to the left of that position crosses any of the connections to words to the right and if, further, none of the words in the English sentence has connections to words on both the left and the right of the position. A set of rifts divides the sentence in which it occurs into a series of *segments*. These segments may, but need not, resemble grammatical phrases.

If a French sentence contains a rift, it is clear that we can construct a translation of the complete sentence by concatenating a translation for the words to the right of the rift with a translation for the words to the left of the rift. Similarly, if a French sentence contains a number of rifts, then we can piece together a translation of the complete sentence from translations of the individual segments. Because of this, we assume that breaking a French sentence at a rift is

less likely to cause a translation error than breaking it elsewhere.

Let $\Pr(e, a|f)$ be the conditional probability of the English sentence e and the alignment a given the French sentence $f = f_1 f_2 \ldots f_M$. For $1 \le i < M$, let $I(i; e, a, f)$ be 1 if there is a rift between f_i and f_{i+1} when f is translated as e with alignment a, and zero otherwise. The probability that f has a rift between f_i and f_{i+1} is given by

$$p(r|i; f) \equiv \sum_{e, a} I(i; e, a, f) \Pr(e, a|f). \qquad (1)$$

Notice that $p(r|i, f)$ depends on f, but not on any translation of it, and can therefore be determined solely from an analysis of f itself.

The Data

We have at our disposal a large collection of French sentences aligned with their English translations [2, 4]. From this collection, we have extracted sentences comprising 27,217,234 potential rift locations as data from which to construct a model for estimating $p(r|i; f)$. Of these locations, we determined 13,268,639 to be rifts and the remaining 13,948,592 not to be rifts. Thus, if we are asked whether a particular position is or is not a rift, but are given no information about the position, then our uncertainty as to the answer will be 0.9995 bits. We were surprised that this entropy should be so great.

In the examples below, which we have chosen from our aligned data, the rifts are indicated by carets appearing between some of the words.

1. La$_\wedge$réponse$_\wedge$à$_\wedge$la$_\wedge$question #2$_\wedge$est$_\wedge$oui$_\wedge$.

2. Ce$_\wedge$chiffre$_\wedge$compris$_\wedge$la rémunération
   du temps supplémentaire$_\wedge$.

3. La$_\wedge$Société du crédit agricole$_\wedge$
   fait savoir$_\wedge$ce qui suit:

The exact positions of the rifts in these sentences depends on the English translation with which they are aligned. For the first sentence above, the Hansard English is *The answer to part two is yes.* If, instead, it had been *For part two, yes is the answer*, then the only rift in the sentence would have appeared immediately before the final punctuation.

The Decision Tree

Brown *et al.* [3] describe a method for assigning sense labels to words in French sentences. Their idea is this. Given a French word f, find a series of yes-no questions about the context in which it occurs so that knowing the answers to these questions reduces the entropy of the translation of f. They assume that the sense of f can be determined from an examination of the French words in the vicinity of f. They refer to these words as informants and limit their search to questions of the form *Is some particular informant in a particular subset of the French vocabulary*. The set of possible answers to these questions can be displayed as a tree, the leaves of which they take to correspond to the senses of f.

We have adapted this technique to construct a decision tree for estimating $p(r|i,\mathbf{f})$. Changing any of the words in $\mathbf{f}$ may affect $p(r|i,\mathbf{f})$, but we consider only its dependence on f_{i-1} through f_{i+2}, the four words closest to the location of the potential rift, and on the parts of speech of these words. We treat each of these eight items as a candidate informant. For each of the 27,217,234 training locations, we created a record of the form $v_1 \, v_2 \, v_3 \, v_4 \, v_5 \, v_6 \, v_7 \, v_8 \, b$, where v_s is the value of the informant at site s and b is 1 or 0 according as the location is or is not a rift. Using 20,000,000 of these records as data, we have constructed a binary decision tree with a total of 245 leaves.

Each of the 244 internal nodes of this tree has associated with it one of the eight informant sites, a subset of the informant vocabulary for that site, a left son, and a right son. For node n, we represent this information by the quadruple $\langle s(n), \mathcal{S}(n), l(n), r(n) \rangle$. Given any location in a French sentence, we construct $v_1 \, v_2 \, v_3 \, v_4 \, v_5 \, v_6 \, v_7 \, v_8$ and assign the location to a leaf as follows.

1. Set a to the root node.
2. If a is a leaf, then assign the location to a and stop.
3. If $v_{s(a)} \in \mathcal{S}(a)$, then set a to $l(a)$, otherwise set a to $r(a)$.
4. Go to step 2.

We call this process *pouring* the data down the tree. We call the series of values that a takes the *path* of

the data down the tree. Each path begins at the root node and ends at a leaf node.

We used this algorithm to pour our 27,217,234 training locations down the tree. We estimate $p(r|i,\mathbf{f})$ at a leaf to be the fraction of these training locations at the leaf for which $b = 1$. In a similar manner, we can estimate $p(r|i,\mathbf{f})$ at each of the internal nodes of the tree. We write $p_c(n)$ for the estimate of $p(r|i,\mathbf{f})$ obtained in this way at node n. The average entropy of b at the leaves is 0.7669 bits. Thus, by using the decision tree, we can reduce the entropy of b for training data by 0.2326 bits.

To warrant our tree against idiosyncrasies in the training data, we used an additional 528,509 locations as data for smoothing the distributions at the leaves. We obtain a smooth estimate, $p(n)$, of $p(r|i,\mathbf{f})$ at each node as follows. At the root, we take $p(n)$ to equal $p_c(n)$. At all other nodes, we define

$$p(n) = \lambda(b_n)p_c(n) + (1 - \lambda(b_n))p(\text{the parent of } n),$$
(2)

where b_n is one of fifty buckets associated with a node according to the count of training locations at the node. Bucket 1 is for counts of 0 and 1, bucket 50 is for counts equal to or greater than 1,000,000, and for $1 < i < 50$, bucket i is for counts greater than or equal to $x_i - \sigma\sqrt{x_i}$ and less than $x_i + \sigma\sqrt{x_i}$, with $x_2 - \sigma\sqrt{x_2} = 2$, $x_{49} + \sigma\sqrt{x_{49}} = 1,000,000$, and $x_i + \sigma\sqrt{x_i} = x_{i+1} - \sigma\sqrt{x_{i+1}}$ for $1 < i < 49$. Here, $x_2 = 438$, and $\sigma = 21$.

Segmenting

Let $t(l)$ be the expected time required by our system to translate a sequence of l French words. We can estimate $t(l)$ for small values of l by using our system to translate a number of sentences of length l. If we break $\mathbf{f}$ into $m+1$ pieces by splitting it between f_{i_1} and f_{i_1+1}, between f_{i_2} and f_{i_2+1}, and so on, finishing with a split between f_{i_m} and f_{i_m+1}, $1 \leq i_1 < i_2 < \cdots < i_m < M$, then the expected time to translate all of the pieces is $t(i_1)+t(i_2-i_1)+\cdots+t(i_m-i_{m-1})+t(M-i_m)$. Translation accuracy will be largely unaffected exactly when each split falls on a rift. Assuming that rifts occur independently of one another, the probability of this event is $\prod_{k=1}^m p(r|i_k,\mathbf{f})$. We define the utility, $S_\alpha(\mathbf{i},\mathbf{f})$, of a split $\mathbf{i} = (i_1, i_2, \ldots, i_m)$ for $\mathbf{f}$ by

$$S_\alpha(\mathbf{i},\mathbf{f}) = \alpha \sum_{k=1}^{m} \log p(r|i_k,\mathbf{f}) -$$

$$(1 - \alpha)(t(i_1) + t(i_2 - i_1) +$$
$$\cdots + t(i_m - i_{m-1}) + t(M - i_m)).$$

Here, α is a parameter weighing accuracy against translation time: when α is near 1, we favor accuracy (and, hence, few segments) at the expense of translation time; when α is near zero, we favor translation time (and, hence, many segments) at the expense of accuracy.

Given a French sentence $\mathbf{f}$ and the decision tree mentioned above for approximating $p(r|i, \mathbf{f})$, it is straightforward using dynamic programming to find the split that maximizes S_α.

If we approximate $t(l)$ to be zero for l less than some threshold and infinite for l equal to or greater than that threshold, then we can discard α. Our utility becomes simply

$$S(\mathbf{i}, \mathbf{f}) = \sum_{k=1}^{m} \log p(r|i_k, \mathbf{f})$$

provided all of the segments are less than the threshold. If the length of any segment is equal to or greater than the threshold, then the utility is $-\infty$.

Decoding

In the absence of segmentation, we employ an analysis-transfer-synthesis paradigm in our decoder as described in detail by Brown et al. [5]. We have insinuated the segmenter into the system between the analysis and the transfer phases of our processing. The analysis operation, therefore, is unaffected by the presence of the segmenter. We have also modified the transfer portion of the decoder so as to investigate only those translations that are consistent with the segmented input, but have otherwise left it alone. As a result, we get the benefit of the English language model across segment boundaries, but save time by not considering the great number of translations that are not consistent with the segmented input.

Results

To test the usefulness of segmenting, we decoded 400 short sentences four different ways. We compiled the results in Table 1, where: *Tree* is a shorthand for segmentation using the tree described above with a threshold of 7; *Every 5* is a shorthand for segments made regularly after every five words; *Every 4* is a shorthand for segments made regularly after every four words; and *None* is a shorthand for using no segmentation at all. We see from the first line of the table that the decoder performed somewhat better with segmentation as determined by the decision tree. If we carried out an exhaustive search, this could not happen, but because our search is suboptimal it is possible for the various shortcuts that we have taken to interact in such a way as to make the result better with segmentation than without. The result with the decision tree is clearly superior to the results obtained with either of the rigid segmentation schemes.

In Table 2, we show the decoding time in minutes for the four decoders. Using the segmentation tree, the decoder is about 41% faster than without it. We use a trigram language model to provide the *a priori* probability for English sentences. This means that the translation of one segment may depend on the result of the immediately preceding segment, but should not be much affected by the translation of any earlier segment provided that segments average more than two words in length. Because of this, we expect translation time with the segmenter to grow approximately linearly with sentence length, while translation time without the segmenter grows much more rapidly. Therefore, we anticipate that the benefit of segmenting to decoding speed will be greater for longer sentences.

	Better	Worse	Equal
Tree *vs.* None	8	12	380
Every 5 *vs.* None	16	55	329
Every 4 *vs.* None	11	61	328
Tree *vs.* Every 5	54	18	328
Tree *vs.* Every 4	59	11	330

Table 1. Comparison of segmentation schemes

Method	Translation time (in minutes)	Average Segment Length
None	8716	9.35
Every 5	4628	4.12
Tree	5124	3.67
Every 4	4414	3.44

Table 2. Translation times and segment lengths

REFERENCES

[1] P. F. Brown, J. Cocke, S. A. DellaPietra, V. J. DellaPietra, F. Jelinek, J. D. Lafferty, R. L. Mer-

cer, and P. S. Roossin, "A statistical approach to machine translation," *Computational Linguistics*, vol. 16, pp. 79–85, June 1990.

[2] P. F. Brown, J. C. Lai, and R. L. Mercer, "Aligning sentences in parallel corpora," in *Proceedings 29th Annual Meeting of the Association for Computational Linguistics*, (Berkeley, CA), pp. 169–176, June 1991.

[3] P. F. Brown, S. A. DellaPietra, V. J. DellaPietra, and R. L. Mercer, "Word sense disambiguation using statistical methods," in *Proceedings 29th Annual Meeting of the Association for Computational Linguistics*, (Berkeley, CA), pp. 265–270, June 1991.

[4] P. F. Brown, S. A. DellaPietra, V. J. DellaPietra, and R. L. Mercer, "The mathematics of machine translation: Parameter estimation." Submitted to Computational Linguistics, 1991.

[5] P. F. Brown, S. A. DellaPietra, V. J. DellaPietra, J. Lafferty, and R. L. Mercer, "Analysis, statistical transfer, and synthesis in machine translation." Submitted to TMI-92, Fourth International Conference on Theoretical and Methodological Issues in Machine Translation, 1992.