Squibs

Does GIZA++ Make Search Errors?

Sujith Ravi*
University of Southern California

Kevin Knight**
University of Southern California

Word alignment is a critical procedure within statistical machine translation (SMT). Brown et al. (1993) have provided the most popular word alignment algorithm to date, one that has been implemented in the GIZA (Al-Onaizan et al. 1999) and GIZA++ (Och and Ney 2003) software and adopted by nearly every SMT project. In this article, we investigate whether this algorithm makes search errors when it computes Viterbi alignments, that is, whether it returns alignments that are sub-optimal according to a trained model.

1. Background

Word alignment is the problem of annotating a bilingual text with links connecting words that have the same meanings. Brown et al. (1993) align an English/French sentence pair by positing a probabilistic model by which an English sentence is translated into French.[1] The model provides a set of non-deterministic choices. When a particular sequence of choices is applied to an English input sentence $e_1 ... e_l$, the result is a particular French output sentence $f_1 ... f_m$. In the Brown et al. models, a decision sequence also implies a specific word alignment vector $a_1 ... a_m$. We say $a_j = i$ when French word f_j was produced by English word e_i during the translation. Here is a sample sentence pair (**e**, **f**) and word alignment **a**:

e: NULL_0 Mary_1 did_2 not_3 slap_4 the_5 green_6 witch_7

f: Mary_1 no_2 dió_3 una_4 bofetada_5 a_6 la_7 bruja_8 verde_9

a: [1 3 4 5 5 0 5 7 6]

Notice that the English sentence contains a special NULL word (e_0) that generates "spurious" target words (in this case, a_6). The Brown et al. (1993) models are many-to-one, meaning that each English word can produce several French children, but each

* Information Sciences Institute, University of Southern California, 4676 Admiralty Way, Marina del Rey, CA 90292. E-mail: sravi@isi.edu.
** Information Sciences Institute, University of Southern California, 4676 Admiralty Way, Marina del Rey, CA 90292. E-mail: knight@isi.edu.

1 We follow standard convention in using "English" and "French" simply as shorthand for "source" and "target". In fact, we use English/Spanish examples in this article.

Submission received: 16 December 2009; accepted for publication: 7 April 2010.

French word has only one English parent. This is why we can represent an alignment as a vector $a_1 \ldots a_m$. There are $(l+1)^m$ ways to align $(\mathbf{e}, \mathbf{f})$. For Brown et al. the goal of word alignment is to find the alignment $\mathbf{a}$ that is most likely, given a sentence pair $(\mathbf{e}, \mathbf{f})$:

$$\text{argmax}_{\mathbf{a}} \, \text{P}(\mathbf{a}|\mathbf{e}, \mathbf{f}) = \text{argmax}_{\mathbf{a}} \text{P}(\mathbf{a}, \mathbf{f}|\mathbf{e}) \tag{1}$$

2. IBM Model 3

Supplied with a formula for $\text{P}(\mathbf{a}|\mathbf{e}, \mathbf{f})$, we can search through the $(l+1)^m$ alignments for the highest-scoring one. Brown et al. (1993) come up with such a formula by first positing this generative story (IBM Model 3):

Given an English sentence $e_1 \ldots e_l$:

1. Choose a fertility ϕ_i for each English word e_i, according to the distribution $n(\phi_i|e_i)$.
2. Let $m' = \sum_{i=1 \ldots l} \phi_i$.
3. Choose a number ϕ_0 of "spurious" French words, by doing the following m' times: with probability p_1, increment ϕ_0 by one.
4. Let $m = m' + \phi_0$.
5. Choose a French translation τ_{ik} for each English word (including e_0) and fertility value, according to the distribution $t(\tau_{ik}|e_i)$.
6. Choose a French position j for each τ_{ik} $(i > 0)$, according to $d(j|i, l, m)$. If the same j is chosen twice, fail the procedure.
7. Place each of the ϕ_0 spuriously generated words, one by one, into vacant positions in the English string, according to uniform probability.
8. Output the French string $f_1 \ldots f_m$, where f_j is the French word τ_{ik} that was placed into position j in Step 6 or 7.
9. Output alignment $a_1 \ldots a_m$, where a_j is the English position i from that same τ_{ik}.

Different decision sequences can result in the same outputs $\mathbf{f}$ and $\mathbf{a}$. With this in mind, Brown et al. provide the following formula:

$$\begin{aligned} \text{P}(\mathbf{a}, \mathbf{f}|\mathbf{e}) = & \prod_{j=1}^{m} t(f_j|e_{a_j}) \cdot \prod_{i=1}^{l} n(\phi_i|e_i) \cdot \prod_{a_j \neq 0, j=1}^{m} d(j|a_j, l, m) \\ & \cdot \prod_{i=0}^{l} \phi_i! \cdot \frac{1}{\phi_0!} \cdot \binom{m - \phi_0}{\phi_0} \cdot p_1^{\phi_0} \cdot p_0^{m-2\phi_0} \end{aligned} \tag{2}$$

Note that terms $\phi_0 \ldots \phi_l$ are only shorthand, as their values are completely determined by the alignment $a_1 \ldots a_m$.

3. Finding the Viterbi Alignment

We assume that probability tables n, t, d, and p have already been learned from data, using the EM method described by Brown et al. (1993). We are concerned solely with the problem of then finding the best alignment for a given sentence pair (**e**, **f**) as described in Equation 1. Brown et al. were unable to discover a polynomial time algorithm for this problem, which was in fact subsequently shown to be NP-complete (Udupa and Maji 2006). Brown et al. therefore devise a hill-climbing algorithm. This algorithm starts with a reasonably good alignment (Viterbi IBM Model 2, computable in quadratic time), after which it greedily executes small changes to the alignment structure, gradually increasing P(**a**, **f**|**e**). The small changes consist of *moves*, in which the value of some a_j is changed, and *swaps*, in which a pair a_j and a_k exchange values. At each step in the greedy search, all possible moves and swaps are considered, and the one which increases P(**a**, **f**|**e**) the most is executed. When P(**a**, **f**|**e**) can no longer be improved, the search halts.[2]

Our question is whether this hill-climbing algorithm, as implemented in GIZA++, makes search errors when it tries to locate IBM Model 3 Viterbi alignments. To answer this, we built a slow but optimal IBM Model 3 aligner, by casting the problem in the integer linear programming (ILP) framework. Given a sentence pair ($e_1 ... e_l$, $f_1 ... f_m$), we set up the following variables:

- Binary link variables $link_{ij}$. We have one such variable for each pair of English and French tokens. If $link_{ij} = 1$, then there is an alignment link between e_i and f_j. The English NULL word is represented by $i = 0$.
- Binary fertility variables $fert_{ik}$. We have one variable for each English token paired with an integer fertility value 0..9. If $fert_{ik} = 1$, then token e_i has fertility k.

To ensure that values we assign to variables are legal and self-consistent, we introduce several constraints. First, we require that for each j, all $link_{xj}$ values sum to one. This means each French token has exactly one alignment link, as required by IBM Model 3. Second, we require that for each i, all $fert_{ix}$ values sum to one, so that each English token has a unique fertility. Third, we require that link and fertility variable assignments be consistent with one another: for each i, the sum of $link_{ix}$ variables equals $1 \cdot fert_{i1} + 2 \cdot fert_{i2} + \ldots + 9 \cdot fert_{i9}$.[3]

We then define an objective function whose minimization corresponds to finding the Viterbi IBM Model 3 alignment. Figure 1 presents the components of the objective function, alongside counterparts from the P(**a**, **f**|**e**) formula previously given. Note that the coefficients for the objective function are derived from the already-learned probability tables n, d, t, and p.

For each sentence pair in our test set, we create and solve an ILP problem. Figure 2 demonstrates this for a simple example. The reader can extract the optimal alignment from the alignment variables chosen in the ILP solution.

2 Brown et al. (1993) describe a variation called *pegging*, which carries out multiple additional greedy hill-climbs, each with a different IBM Model 2 Viterbi link fixed (pegged) for the duration of the hill-climb. In practice, pegging is slow, and the vast majority of GIZA++ users do not employ it.

3 The default configuration of GIZA++ includes this same fertility cap of 9, though the Brown et al. (1993) description does not.

IBM Model 3 decoding

ILP objective function

Choose values for $link_{ij}$ and $fert_{ik}$ variables to Minimize:

$$\sum_{i=0\ldots l, j=1\ldots m} link_{ij} \cdot [\, -log_2\, t(f_j|e_i)\,]$$

$$+ \sum_{i=1\ldots l, k=0\ldots 9} fert_{ik} \cdot [\, -log_2\, n(k|e_i)\,]$$

$$+ \sum_{i=1\ldots l, j=1\ldots m} link_{ij} \cdot [\, -log_2\, d(j|i,l,m)\,]$$

$$+ \sum_{i=1\ldots l, k=0\ldots 9} fert_{ik} \cdot [\, -log_2\, (k!)\,]$$

$$+ \sum_{k=0\ldots 9} fert_{0k} \cdot [\, -log_2\, (\frac{1}{k!}) \, - log_2 \binom{m-k}{k}$$

$$-k \cdot \; log_2\,(p_1) \; - (m-2k) \cdot \; log_2\,(p_0)\,]$$

IBM Model 3 formula

Choose an alignment **a** to Maximize:

$$\prod_{j=1}^{m} t(f_j|e_{a_j})$$

$$\cdot \prod_{i=1}^{l} n(\phi_i|e_i)$$

$$\cdot \prod_{a_j \neq 0, j=1}^{m} d(j|a_j, l, m)$$

$$\cdot \prod_{i=0}^{l} \phi_i!$$

$$\cdot \frac{1}{\phi_0!} \cdot \binom{m-\phi_0}{\phi_0}$$

$$\cdot p_1^{\phi_0} \cdot p_0^{m-2\phi_0}$$

ILP constraints

1. Single fertility for every English word (including NULL)

$$\forall_i, \sum_{k=0\ldots 9} fert_{ik} = 1$$

2. Single alignment link for every foreign word

$$\forall_j, \sum_{i=0\ldots l} link_{ij} = 1$$

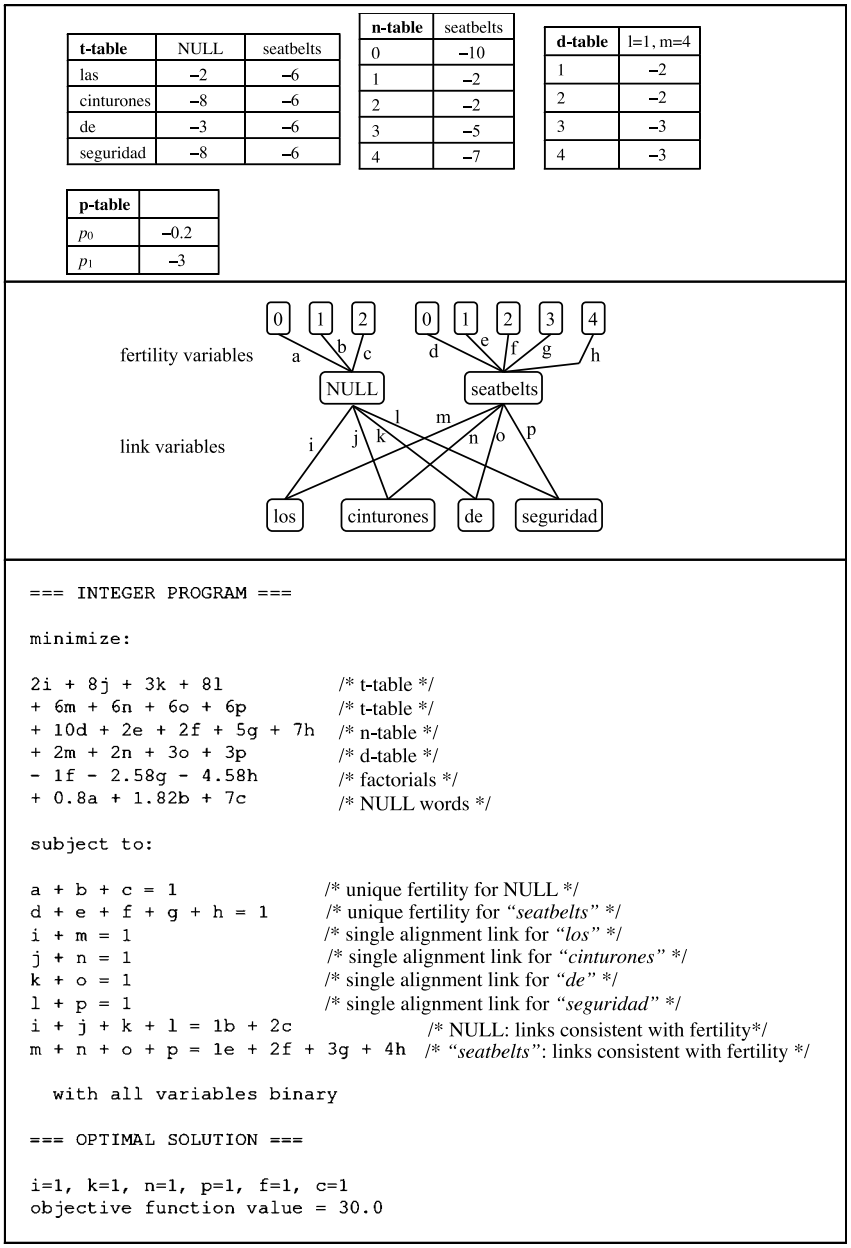
3. Links should be consistent with chosen fertility

$$\forall_i, \sum_{j=1\ldots m} link_{ij} = \sum_{k=0\ldots 9} k \cdot fert_{ik}$$

Figure 1
Objective function and constraints (left) for the ILP formulation that encodes the problem of selecting the Viterbi alignment for a sentence pair under IBM Model 3. Components of the ILP objective function are paired with counterparts (right) from the Model 3's P(**a**, **f**|**e**) formula.

4. Experiments

For Chinese/English experiments, we run GIZA++ training on 101,880 sentence pairs. We evaluate Viterbi alignments on a smaller test set of 1,880 sentence pairs with manual alignments. For Arabic/English, we train on 300,000 sentence pairs and test on 2,000. In tests, we compare GIZA++ Viterbi alignments (based on greedy hill-climbing) with optimal ILP alignments. We use CPLEX to solve our ILP problems.



t-table	NULL	seatbelts
las	–2	–6
cinturones	–8	–6
de	–3	–6
seguridad	–8	–6

n-table	seatbelts
0	–10
1	–2
2	–2
3	–5
4	–7

d-table	l=1, m=4
1	–2
2	–2
3	–3
4	–3

p-table	
p_0	–0.2
p_1	–3


```
=== INTEGER PROGRAM ===

minimize:

2i + 8j + 3k + 8l            /* t-table */
+ 6m + 6n + 6o + 6p          /* t-table */
+ 10d + 2e + 2f + 5g + 7h    /* n-table */
+ 2m + 2n + 3o + 3p          /* d-table */
- 1f - 2.58g - 4.58h         /* factorials */
+ 0.8a + 1.82b + 7c          /* NULL words */

subject to:

a + b + c = 1               /* unique fertility for NULL */
d + e + f + g + h = 1       /* unique fertility for "seatbelts" */
i + m = 1                   /* single alignment link for "los" */
j + n = 1                   /* single alignment link for "cinturones" */
k + o = 1                   /* single alignment link for "de" */
l + p = 1                   /* single alignment link for "seguridad" */
i + j + k + l = 1b + 2c             /* NULL: links consistent with fertility*/
m + n + o + p = 1e + 2f + 3g + 4h   /* "seatbelts": links consistent with fertility */

  with all variables binary

=== OPTIMAL SOLUTION ===

i=1, k=1, n=1, p=1, f=1, c=1
objective function value = 30.0
```

Figure 2
Illustration of ILP-based optimal alignment of a single sentence pair. Already-trained log probabilities are shown at the top of the figure. In the middle is a schematic of variables introduced for the English/Spanish sentence pair *seatbelts / los cinturones de seguridad*. At the bottom is the ILP formulation and its solution.

Figure 3 compares the results from GIZA++ alignment with optimal ILP alignment for different language pairs and alignment directions, and for unions of uni-directional alignments. We measure the rate at which GIZA++ makes search errors, and we compute alignment F-scores for the various testing conditions. We conclude that although

Language pair and direction	Sentences for GIZA++ training	Subset of sentences for alignment evaluation (LDC-aligned)	Percent of sentences on which GIZA++ Model 3 Viterbi alignment computation makes a search error	GIZA++ Viterbi alignment quality (F-score)	ILP (optimal) Viterbi alignment quality (F-score)
Eng → Chi	101,880	1,880	8.7 (164/1,880)	57.22	57.20
Chi → Eng	101,880	1,880	14.8 (278/1,880)	65.49	65.49
(Eng → Chi) UNION (Chi → Eng)				66.53	66.48
Eng → Ara	300,000	2,000	5.1 (102/2,000)	43.59	43.61
Ara → Eng	300,000	2,000	4.5 (90/2,000)	55.21	55.17
(Eng → Ara) UNION (Ara → Eng)				54.31	54.30

Figure 3
Comparison of GIZA++ Viterbi alignments (based on greedy hill-climbing) with optimal ILP alignments for different language pairs and alignment directions in terms of alignment quality (F-score). The figure also shows the alignment F-scores for UNION alignments that combine alignment links from both directions. The fourth column shows the percentage of sentences on which GIZA++ makes search errors in comparison to optimal ILP alignments.

GIZA++ makes search errors on 5–15% of sentence pairs, these errors do not contribute to an overall loss in alignment task accuracy, as measured by F-score.

Focusing on sentence pairs where GIZA++ makes a search error, we plot the average difference in log model scores between GIZA++ and ILP Viterbi alignments in Figure 4. We notice a positive correlation between sentence length and the search error

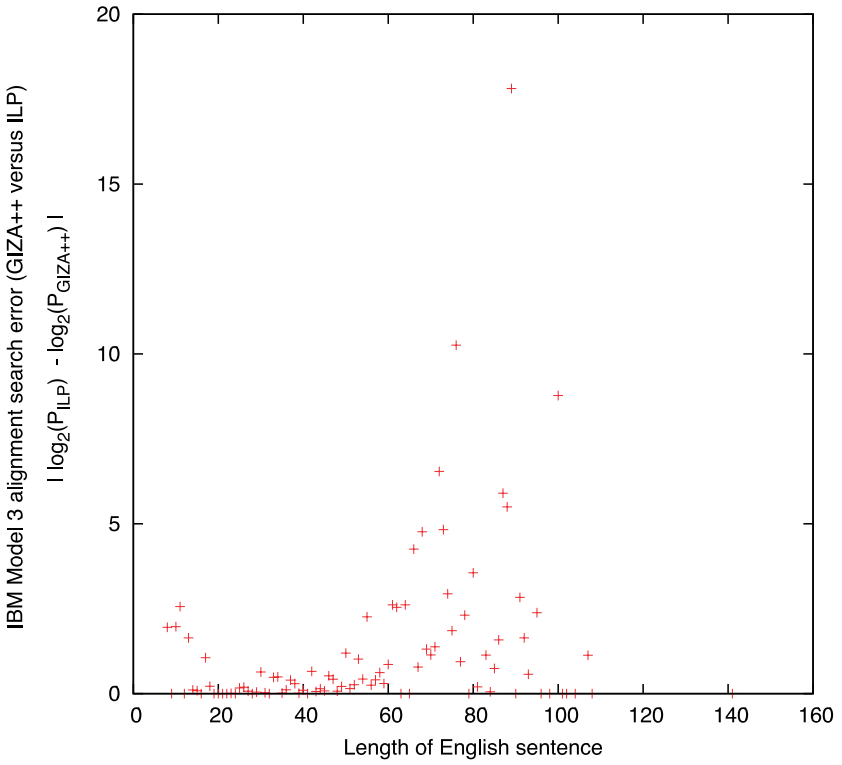


Figure 4
Average difference in log model scores between GIZA++ and ILP alignments at different English sentence lengths for English/Chinese alignment. Points in the plot that appear to be on the *x*-axis actually lie just above it.

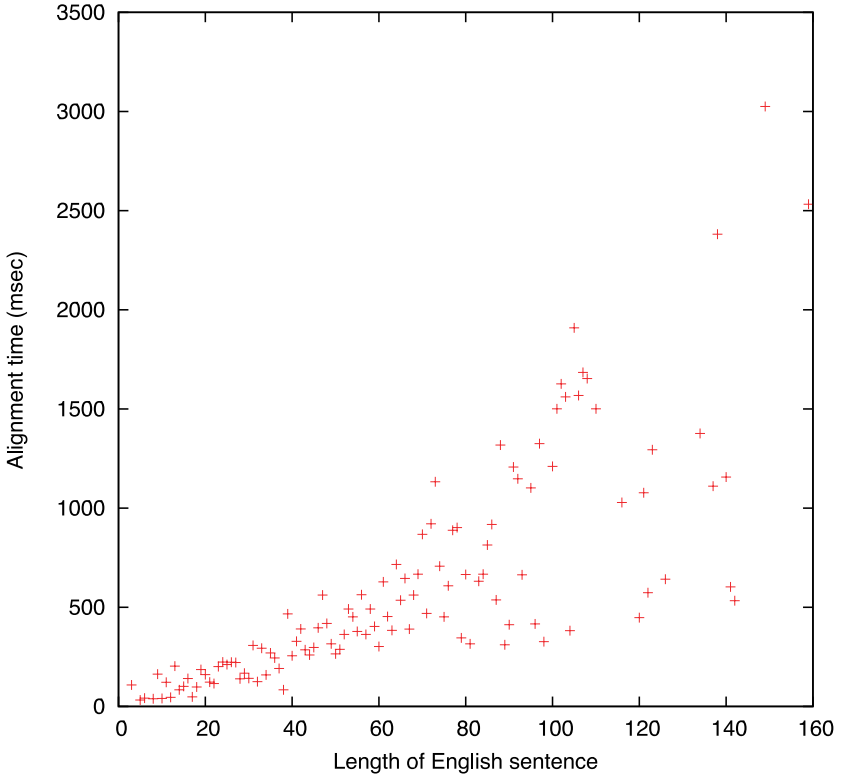


Figure 5
Average time (msec) taken by the ILP aligner at different English sentence lengths for English/Chinese alignment. The experiments were run on a single machine with a 64-bit, 2.4 GHz AMD Opteron 850 processor.

gap between GIZA++ and ILP scores. As we move to longer sentences, the alignment procedure becomes harder and GIZA++ makes more errors. Finally, Figure 5 plots the time taken for ILP alignment at different sentence lengths showing a positive correlation as well.

5. Discussion

We have determined that GIZA++ makes few search errors, despite the heuristic nature of the algorithm. These search errors do not materially affect overall alignment accuracy. In practice, this means that researchers should not spend time optimizing this particular aspect of SMT systems.

Search errors can occur in many areas of SMT. The area that has received the most attention is runtime decoding/translation. For example, Germann et al. (2001) devise an optimal ILP decoder to identify types of search errors made by other decoders. A second area (this article) is finding Viterbi alignments, given a set of alignment parameter values. A third area is actually learning those parameter values. Brown et al.'s (1993) EM learning algorithm aims to optimize the probability of the French side of the parallel corpus given the English side. For Model 3 and above, Brown et al. collect parameter counts over subsets of alignments, instead of over all alignments. These subsets, like Viterbi alignments, are generated heuristically, and it may be that true n-best lists of

alignments would yield better counts and better overall parameter values. Of course, even if we were able to collect accurate counts, EM is not guaranteed to find a global optimum, which provides further opportunity for search errors. We leave the problem of search errors in alignment training to future study.

Acknowledgments

This work was supported by NSF grant 0904684 and DARPA GALE Contract Number HR0011-06-C-0022.

References

Al-Onaizan, Yaser, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Franz Josef Och Dan Melamed, David Purdy, Noah Smith, and David Yarowsky. 1999. Statistical machine translation. Technical report, Johns Hopkins University.

Brown, Peter, Vincent Della Pietra, Stephen Della Pietra, and Robert Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Germann, Ulrich, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2001. Fast decoding and optimal decoding for machine translation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 228–235, Toulouse.

Och, Franz Josef and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Udupa, Raghavendra and Hemanta K. Maji. 2006. Computational complexity of statistical machine translation. In *Proceedings of the Conference of the European Chapter of the Association of Computational Linguistics (EACL)*, pages 25–32, Trento.