Fast Inference in Phrase Extraction Models with Belief Propagation

David Burkett and **Dan Klein**
Computer Science Division
University of California, Berkeley
{dburkett,klein}@cs.berkeley.edu

Abstract

Modeling overlapping phrases in an alignment model can improve alignment quality but comes with a high inference cost. For example, the model of DeNero and Klein (2010) uses an ITG constraint and beam-based Viterbi decoding for tractability, but is still slow. We first show that their model can be approximated using structured belief propagation, with a gain in alignment quality stemming from the use of marginals in decoding. We then consider a more flexible, non-ITG matching constraint which is less efficient for exact inference but *more* efficient for BP. With this new constraint, we achieve a relative error reduction of 40% in F_5 and a 5.5x speed-up.

1 Introduction

Modern statistical machine translation (MT) systems most commonly infer their transfer rules from word-level alignments (Koehn et al., 2007; Li and Khudanpur, 2008; Galley et al., 2004), typically using a deterministic heuristic to convert these to phrase alignments (Koehn et al., 2003). There have been many attempts over the last decade to develop model-based approaches to the phrase alignment problem (Marcu and Wong, 2002; Birch et al., 2006; DeNero et al., 2008; Blunsom et al., 2009). However, most of these have met with limited success compared to the simpler heuristic method. One key problem with typical models of phrase alignment is that they choose a single (latent) segmentation, giving rise to undesirable modeling biases (DeNero et al., 2006) and reducing coverage, which in turn reduces translation quality (DeNeefe et al., 2007; DeNero et al., 2008). On the other hand, the extraction heuristic identifies many overlapping options, and achieves high coverage.

In response to these effects, the recent phrase alignment work of DeNero and Klein (2010) models *extraction sets*: collections of overlapping phrase pairs that are consistent with an underlying word alignment. Their extraction set model is empirically very accurate. However, the ability to model overlapping – and therefore non-local – features comes at a high computational cost. DeNero and Klein (2010) handle this in part by imposing a structural ITG constraint (Wu, 1997) on the underlying word alignments. This permits a polynomial-time algorithm, but it is still $O(n^6)$, with a large constant factor once the state space is appropriately enriched to capture overlap. Therefore, they use a heavily beamed Viterbi search procedure to find a reasonable alignment within an acceptable time frame. In this paper, we show how to use belief propagation (BP) to improve on the model's ITG-based structural formulation, resulting in a new model that is simultaneously faster and more accurate.

First, given the model of DeNero and Klein (2010), we decompose it into factors that admit an efficient BP approximation. BP is an inference technique that can be used to efficiently approximate posterior marginals on variables in a graphical model; here the marginals of interest are the phrase pair posteriors. BP has only recently come into use in the NLP community, but it has been shown to be effective in other complex structured classification tasks, such as dependency parsing (Smith and Eisner, 2008). There has also been some prior success in using BP for both discriminative (Niehues and Vogel, 2008) and generative (Cromières and Kurohashi, 2009) word alignment models.

By aligning all phrase pairs whose posterior under BP exceeds some fixed threshold, our BP approximation of the model of DeNero and Klein (2010) can

achieve a comparable phrase pair F_1. Furthermore, because we have posterior marginals rather than a single Viterbi derivation, we can explicitly force the aligner to choose denser extraction sets simply by lowering the marginal threshold. Therefore, we also show substantial improvements over DeNero and Klein (2010) in recall-heavy objectives, such as F_5.

More importantly, we also show how the BP factorization allows us to relax the ITG constraint, replacing it with a new set of constraints that permit a wider family of alignments. Compared to ITG, the resulting model is *less* efficient for exact inference (where it is exponential), but *more* efficient for our BP approximation (where it is only quadratic). Our new model performs even better than the ITG-constrained model on phrase alignment metrics while being faster by a factor of 5.5x.

2 Extraction Set Models

Figure 1 shows part of an aligned sentence pair, including the word-to-word alignments, and the extracted phrase pairs licensed by those alignments. Formally, given a sentence pair $(\mathbf{e}, \mathbf{f})$, a word-level alignment a is a collection of links between target words e_i and source words f_j. Following past work, we further divide word links into two categories: sure and possible, shown in Figure 1 as solid and hatched grey squares, respectively. We represent a as a grid of ternary word link variables a_{ij}, each of which can take the value *sure* to represent a sure link between e_i and f_j, *poss* to represent a possible link, or *off* to represent no link.

An extraction set π is a set of aligned phrase pairs to be extracted from $(\mathbf{e}, \mathbf{f})$, shown in Figure 1 as green rounded rectangles. We represent π as a set of boolean variables $\pi_{ghk\ell}$, which each have the value *true* when the target span $[g, h]$ is phrase-aligned to the source span $[k, \ell]$. Following previous work on phrase extraction, we limit the size of π by imposing a phrase length limit d: π only contains a variable $\pi_{ghk\ell}$ if $h - g < d$ and $\ell - k < d$.

There is a deterministic mapping $\pi(a)$ from a word alignment to the extraction set licensed by that word alignment. We will briefly describe it here, and then present our factorized model.

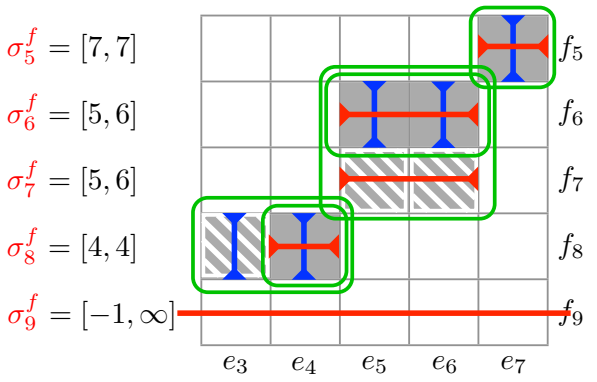


Figure 1: A schematic representation of part of a sentence pair. Solid grey squares indicate sure links (e.g. a_{48} = *sure*), and hatched squares possible links (e.g. a_{67} = *poss*). Rounded green rectangles are extracted phrase pairs (e.g. π_{5667} = *true*). Target spans are shown as blue vertical lines and source spans as red horizontal lines. Because there is a sure link at a_{48}, $\sigma_8^f = [4, 4]$ does not include the possible link at a_{38}. However, f_7 only has possible links, so $\sigma_7^f = [5, 6]$ is the span containing those. f_9 is null-aligned, so $\sigma_9^f = [-1, \infty]$, which blocks all phrase pairs containing f_9 from being extracted.

2.1 Extraction Sets from Word Alignments

The mapping from a word alignment to the set of licensed phrase pairs $\pi(a)$ is based on the standard rule extraction procedures used in most modern statistical systems (Koehn et al., 2003; Galley et al., 2006; Chiang, 2007), but extended to handle possible links (DeNero and Klein, 2010). We start by using a to find a projection from each target word e_i onto a source *span*, represented as blue vertical lines in Figure 1. Similarly, source words project onto target spans (red horizontal lines in Figure 1). $\pi(a)$ contains a phrase pair iff every word in the target span projects within the source span and vice versa. Figure 1 contains an example for $d = 2$.

Formally, the mapping introduces a set of spans σ. We represent the spans as variables whose values are intervals, where $\sigma_i^e = [k, \ell]$ means that the target word e_i projects to the source span $[k, \ell]$. The set of legal values for σ_i^e includes any interval with $0 \leq k \leq \ell < |\mathbf{f}|$ and $\ell - k < d$, plus the special interval $[-1, \infty]$ that indicates e_i is null-aligned. The span variables for source words σ_j^f have target spans $[g, h]$ as values and are defined analogously.

For a set I of positions, we define the *range* func-

tion:

$$\mathit{range}(I) = \begin{cases} [-1, \infty] & I = \emptyset \\ [\min_{i \in I} i, \max_{i \in I} i] & \text{else} \end{cases} \quad (1)$$

For a fixed word alignment a we set the target span variable σ_i^e:

$$\sigma_{i,\text{s}}^e = \mathit{range}(\{j : a_{ij} = \mathit{sure}\}) \quad (2)$$

$$\sigma_{i,\text{p}}^e = \mathit{range}(\{j : a_{ij} \neq \mathit{off}\}) \quad (3)$$

$$\sigma_i^e = \sigma_{i,\text{s}}^e \cap \sigma_{i,\text{p}}^e \quad (4)$$

As illustrated in Figure 1, this sets σ_i^e to the minimal span containing all the source words with a *sure* link to e_i if there are any. Otherwise, because of the special case for *range(I)* when I is empty, $\sigma_{i,\text{s}}^e = [-1, \infty]$, so σ_i^e is the minimal span containing all *poss*-aligned words. If all word links to e_i are *off*, indicating that e_i is null-aligned, then σ_i^e is $[-1, \infty]$, preventing the alignment of any phrase pairs containing e_i.

Finally, we specify which phrase pairs should be included in the extraction set π. Given the spans σ based on a, $\pi(a)$ sets $\pi_{ghk\ell} = \mathit{true}$ iff every word in each phrasal span projects within the other:

$$\begin{aligned} \sigma_i^e &\subseteq [k, \ell] & \forall i \in [g, h] \\ \sigma_j^f &\subseteq [g, h] & \forall j \in [k, \ell] \end{aligned} \quad (5)$$

2.2 Formulation as a Graphical Model

We score triples (a, π, σ) as the dot product of a weight vector w that parameterizes our model and a feature vector $\phi(a, \pi, \sigma)$. The feature vector decomposes into word alignment features ϕ_a, phrase pair features ϕ_π and target and source null word features $\phi_\emptyset^e$ and $\phi_\emptyset^f$:[1]

$$\begin{aligned} \phi(a, \pi, \sigma) = &\sum_{i,j} \phi_a(a_{ij}) + \sum_{g,h,k,\ell} \phi_\pi(\pi_{ghk\ell}) + \\ &\sum_i \phi_\emptyset^e(\sigma_i^e) + \sum_j \phi_\emptyset^f(\sigma_j^f) \end{aligned} \quad (6)$$

This feature function is exactly the same as that used by DeNero and Klein (2010).[2] However, while they formulated their inference problem as a search for the highest scoring triple (a, π, σ) for an observed sentence pair $(\mathbf{e}, \mathbf{f})$, we wish to derive a conditional probability distribution $p(a, \pi, \sigma | \mathbf{e}, \mathbf{f})$. We do this with the standard transformation for linear models: $p(a, \pi, \sigma | \mathbf{e}, \mathbf{f}) \propto \exp(w \cdot \phi(a, \pi, \sigma))$. Due to the factorization in Eq. (6), this exponentiated form becomes a product of local multiplicative factors, and hence our model forms an undirected graphical model, or Markov random field.

In addition to the scoring function, our model also includes constraints on which triples (a, π, σ) have nonzero probability. DeNero and Klein (2010) implicitly included these constraints in their representation: instead of sets of variables, they used a structured representation that *only* encodes triples (a, π, σ) satisfying both the mapping $\pi = \pi(a)$ and the structural constraint that a can be generated by a block ITG grammar. However, our inference procedure, BP, requires that we represent (a, π, σ) as an assignment of values to a set of variables. Therefore, we must explicitly encode all constraints into the multiplicative factors that define the model. To accomplish this, in addition to the soft scoring factors we have already mentioned, our model also includes a set of hard constraint factors. Hard constraint factors enforce the relationships between the variables of the model by taking a value of 0 when the constraints they encode are violated and a value of 1 when they are satisfied. The full factor graph representation of our model, including both soft scoring factors and hard constraint factors, is drawn schematically in Figure 2.

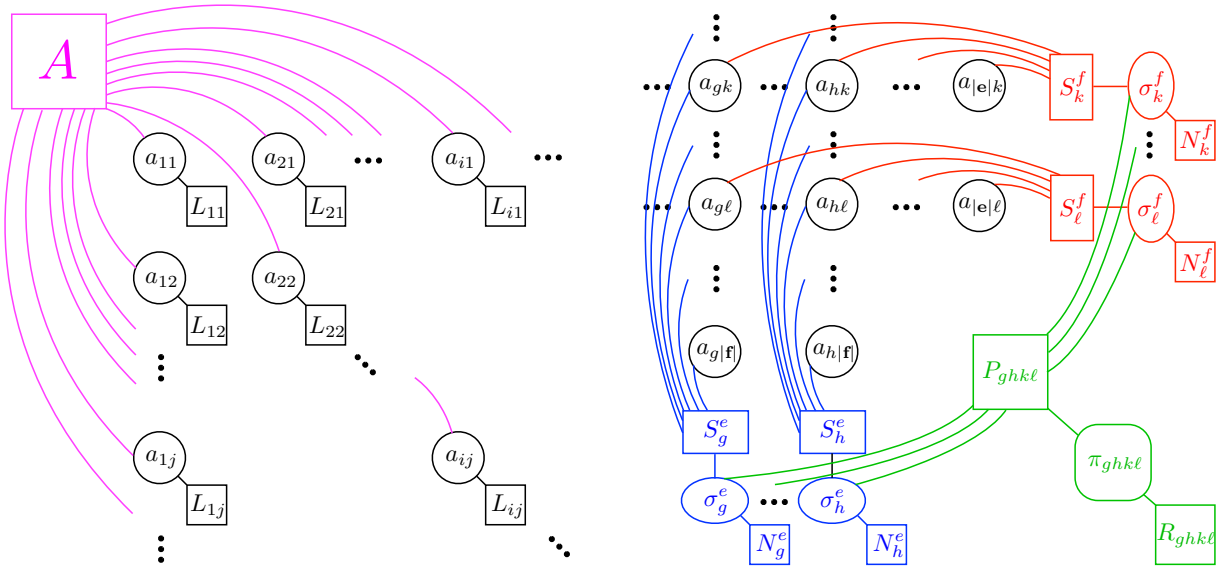
2.2.1 Soft Scoring Factors

The scoring factors all take the form $\exp(w \cdot \phi)$, and so can be described in terms of their respective local feature vectors, ϕ. Depending on the values of the variables each factor depends on, the factor can be active or inactive. Features are only extracted for active factors; otherwise ϕ is empty and the factor produces a value of 1.

SURELINK. Each word alignment variable a_{ij} has a corresponding SURELINK factor L_{ij} to incorporate scores from the features $\phi_a(a_{ij})$. L_{ij} is active whenever $a_{ij} = \mathit{sure}$. $\phi_a(a_{ij})$ includes posteriors from unsupervised jointly trained HMM word alignment models (Liang et al., 2006), dictionary

[1] In addition to the arguments we write out explicitly, all feature functions have access to the observed sentence pair $(\mathbf{e}, \mathbf{f})$.

[2] Although the null word features are not described in DeNero and Klein (2010), all of their reported results include these features (DeNero, 2010).



(a) ITG factor

(b) SPAN and EXTRACT factors

Figure 2: A factor graph representation of the ITG-based extraction set model. For visual clarity, we draw the graph separated into two components: one containing the factors that only neighbor word link variables, and one containing the remaining factors.

and identical word features, a position distortion feature, and features for numbers and punctuation.

PHRASEPAIR. For each phrase pair variable $\pi_{ghk\ell}$, scores from $\phi_\pi(\pi_{ghk\ell})$ come from the factor $R_{ghk\ell}$, which is active if $\pi_{ghk\ell} = \textit{true}$. Most of the model's features are on these factors, and include relative frequency statistics, lexical template indicator features, and indicators for numbers of words and Chinese characters. See DeNero and Klein (2010) for a more comprehensive list.

NULLWORD. We can determine if a word is null-aligned by looking at its corresponding span variable. Thus, we include features from $\phi^e_\emptyset(\sigma^e_i)$ in a factor N^e_i that is active if $\sigma^e_i = [-1, \infty]$. The features are mostly indicators for common words. There are also factors N^f_j for source words, which are defined analogously.

2.2.2 Hard Constraint Factors

We encode the hard constraints on relationships between variables in our model using three families of factors, shown graphically in Figure 2. The SPAN and EXTRACT factors together ensure that $\pi = \pi(a)$. The ITG factor encodes the structural constraint on a.

SPAN. First, for each target word e_i we include a factor S^e_i to ensure that the span variable σ^e_i has a value that agrees with the projection of the word alignment a. As shown in Figure 2b, S^e_i depends on σ^e_i and all the word alignment variables a_{ij} in column i of the word alignment grid. S^e_i has value 1 iff the equality in Eq. (4) holds. Our model also includes a factor S^f_j to enforce the analogous relationship between each σ^f_j and corresponding row j of a.

EXTRACT. For each phrase pair variable $\pi_{ghk\ell}$ we have a factor $P_{ghk\ell}$ to ensure that $\pi_{ghk\ell} = \textit{true}$ iff it is licensed by the span projections σ. As shown in Figure 2b, in addition to $\pi_{ghk\ell}$, $P_{ghk\ell}$ depends on the range of span variables σ^e_i for $i \in [g, h]$ and σ^f_j for $j \in [k, \ell]$. $P_{ghk\ell}$ is satisfied when $\pi_{ghk\ell} = \textit{true}$ and the relations in Eq. (5) all hold, *or* when $\pi_{ghk\ell} = \textit{false}$ and at least one of those relations does *not* hold.

ITG. Finally, to enforce the structural constraint on a, we include a single global factor A that depends on *all* the word link variables in a (see Figure 2a). A is satisfied iff a is in the family of block inverse transduction grammar (ITG) alignments. The block ITG family permits multiple links to be on ($a_{ij} \neq \textit{off}$) for a particular word e_i via terminal block productions, but ensures that every word is

in at most one such terminal production, and that the full set of terminal block productions is consistent with ITG reordering patterns (Zhang et al., 2008).

3 Relaxing the ITG Constraint

The ITG factor can be viewed as imposing two different types of constraints on allowable word alignments a. First, it requires that each word is aligned to at most one relatively short subspan of the other sentence. This is a linguistically plausible constraint, as it is rarely the case that a single word will translate to an extremely long phrase, or to multiple widely separated phrases.[3]

The other constraint imposed by the ITG factor is the ITG reordering constraint. This constraint is imposed primarily for reasons of computational tractability: the standard dynamic program for bitext parsing depends on ITG reordering (Wu, 1997). While this constraint is not dramatically restrictive (Haghighi et al., 2009), it is plausible that removing it would permit the model to produce better alignments. We tested this hypothesis by developing a new model that enforces *only* the constraint that each word align to one limited-length subspan, which can be viewed as a generalization of the at-most-one-to-one constraint frequently considered in the word-alignment literature (Taskar et al., 2005; Cromières and Kurohashi, 2009).

Our new model has almost exactly the same form as the previous one. The only difference is that A is replaced with a new family of simpler factors:

ONESPAN. For each target word e_i (and each source word f_j) we include a hard constraint factor U_i^e (respectively U_j^f). U_i^e is satisfied iff $|\sigma_{i,\mathrm{p}}^e| < d$ (length limit) and either $\sigma_{i,\mathrm{p}}^e = [-1, \infty]$ or $\forall j \in \sigma_{i,\mathrm{p}}^e, a_{ij} \neq \mathit{off}$ (no gaps), with $\sigma_{i,\mathrm{p}}^e$ as in Eq. (3). Figure 3 shows the portion of the factor graph from Figure 2a redrawn with the ONESPAN factors replacing the ITG factor. As Figure 3 shows, there is no longer a global factor; each U_i^e depends only on the word link variables from column i.

[3]Short gaps can be accomodated within block ITG (and in our model are represented as possible links) as long as the total aligned span does not exceed the block size.

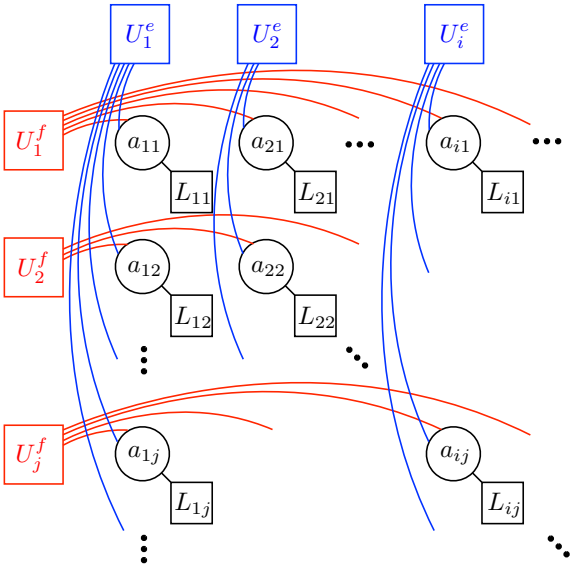


Figure 3: ONESPAN factors

4 Belief Propagation

Belief propagation is a generalization of the well known sum-product algorithm for undirected graphical models. We will provide only a procedural sketch here, but a good introduction to BP for inference in structured NLP models can be found in Smith and Eisner (2008), and Chapters 16 and 23 of MacKay (2003) contain a general introduction to BP in the more general context of message-passing algorithms.

At a high level, each variable maintains a local distribution over its possible values. These local distribution are updated via *messages* passed between variables and factors. For a variable V, $\mathcal{N}(V)$ denotes the set of factors neighboring V in the factor graph. Similarly, $\mathcal{N}(F)$ is the set of variables neighboring the factor F. During each round of BP, messages are sent from each variable to each of its neighboring factors:

$$q_{V\to F}^{(k+1)}(v) \propto \prod_{G\in\mathcal{N}(V), G\neq F} r_{G\to V}^{(k)}(v) \qquad (7)$$

and from each factor to each of its neighboring variables:

$$r_{F\to V}^{(k+1)}(v) \propto \sum_{\mathcal{X}_F, \mathcal{X}_F[V]=v} F(\mathcal{X}_F) \prod_{U\in\mathcal{N}(F), U\neq V} q_{U\to F}^{(k)}(v) \qquad (8)$$

where $\mathcal{X}_F$ is a partial assignment of values to just the variables in $\mathcal{N}(F)$.

Marginal beliefs at time k can be computed by simply multiplying together all received messages and normalizing:

$$b_V^{(k)}(v) \propto \prod_{G \in \mathcal{N}(V)} r_{G \to V}^{(k)}(v) \quad (9)$$

Although messages can be updated according to any schedule, generally one iteration of BP updates each message once. The process iterates until some stopping criterion has been met: either a fixed number of iterations or some convergence metric.

For our models, we say that BP has converged whenever $\sum_{V,v} \left(b_V^{(k)}(v) - b_V^{(k-1)}(v) \right)^2 < \delta$ for some small $\delta > 0$.[4] While we have no theoretical convergence guarantees, it usually converges within 10 iterations in practice.

5 Efficient BP for Extraction Set Models

In general, the efficiency of BP depends directly on the arity of the factors in the model. Performed naïvely, the sum in Eq. (8) will take time that grows exponentially with the size of $\mathcal{N}(F)$. For the soft-scoring factors, which each depend only on a single variable, this isn't a problem. However, our model also includes factors whose arity grows with the input size: for example, explicitly enumerating all assignments to the word link variables that the ITG factor depends on would take $O(3^{n^2})$ time.[5]

To run BP in a reasonable time frame, we need efficient factor-specific propagators that can exploit the structure of the factor functions to compute outgoing messages in polynomial time (Duchi et al., 2007; Smith and Eisner, 2008). Fortunately, all of our hard constraints permit dynamic programs that accomplish this propagation. Space does not permit a full description of these dynamic programs, but we will briefly sketch the intuitions behind them.

SPAN and **ONESPAN.** Marginal beliefs for S_i^e or U_i^e can be computed in $O(nd^2)$ time. The key observation is that for any legal value $\sigma_i^e = [k, \ell]$, S_i^e and U_i^e require that $a_{ij} = \textit{off}$ for all $j \notin [k, \ell]$.[6] Thus, we start by computing the product of all the off beliefs: $\bar{b} = \prod_j q_{a_{ij}}(\textit{off})$. Then, for each of the $O(nd)$ legal source spans $[k, \ell]$ we can efficiently find a joint belief by summing over consistent assignments to the $O(d)$ link variables in that span.

Factor	Runtime	Count	Total
SURELINK	$O(1)$	$O(n^2)$	$O(n^2)$
PHRASEPAIR	$O(1)$	$O(n^2d^2)$	$O(n^2d^2)$
NULLWORD	$O(nd)$	$O(n)$	$O(n^2d)$
SPAN	$O(nd^2)$	$O(n)$	$O(n^2d^2)$
EXTRACT	$O(d^3)$	$O(n^2d^2)$	$O(n^2d^5)$
ITG	$O(n^6)$	1	$O(n^6)$
ONESPAN	$O(nd^2)$	$O(n)$	$O(n^2d^2)$

Table 1: Asymptotic complexity for all factors.

EXTRACT. Marginal beliefs for $P_{ghk\ell}$ can be computed in $O(d^3)$ time. For each of the $O(d)$ target words, we can find the total incoming belief that σ_i^e is within $[k, \ell]$ by summing over the $O(d^2)$ values $[k', \ell']$ where $[k', \ell'] \subseteq [k, \ell]$. Likewise for source words. Multiplying together these per-word beliefs and the belief that $\pi_{ghk\ell} = \textit{true}$ yields the joint belief of a consistent assignment with $\pi_{ghk\ell} = \textit{true}$, which can be used to efficiently compute outgoing messages.

ITG. To build outgoing messages, the ITG factor A needs to compute marginal beliefs for *all* of the word link variables a_{ij}. These can all be computed in $O(n^6)$ time by using a standard bitext parser to run the inside-outside algorithm. By using a normal form grammar for block ITG with nulls (Haghighi et al., 2009), we ensure that there is a 1-1 correspondence between the ITG derivations the parser sums over and word alignments a that satisfy A.

The asymptotic complexity for all the factors is shown in Table 1. The total complexity for inference in each model is simply the sum of the complexities of its factors, so the complexity of the ITG model is $O(n^2d^5 + n^6)$, while the complexity of the relaxed model is just $O(n^2d^5)$. The complexity of exact inference, on the other hand, is exponential in d for the ITG model and exponential in both d and n for the relaxed model.

[4] We set $\delta = 0.001$.

[5] For all asymptotic analysis, we define $n = \max(|\mathbf{e}|, |\mathbf{f}|)$.

[6] For ease of exposition, we assume that all alignments are either *sure* or *off*; the modifications to account for the general case are straightforward.

6 Training and Decoding

We use BP to compute marginal posteriors, which we use at training time to get expected feature counts and at test time for posterior decoding. For each sentence pair, we continue to pass messages until either the posteriors converge, or some maximum number of iterations has been reached.[7] After running BP, the marginals we are interested in can all be computed with Eq. (9).

6.1 Training

We train the model to maximize the log likelihood of manually word-aligned gold training sentence pairs (with L_2 regularization). Because π and σ are determined when a is observed, the model has no latent variables. Therefore, the gradient takes the standard form for loglinear models:

$$\nabla LL = \phi(a, \pi, \sigma) - \sum_{a', \pi', \sigma'} p(a', \pi', \sigma' | \mathbf{e}, \mathbf{f}) \phi(a', \pi', \sigma') - \lambda w \quad (10)$$

The feature vector ϕ contains features on sure word links, extracted phrase pairs, and null-aligned words. Approximate expectations of these features can be efficiently computed using the marginal beliefs $b_{a_{ij}}(\textit{sure})$, $b_{\pi_{ghk\ell}}(\textit{true})$, and $b_{\sigma_i^e}([-1, \infty])$ and $b_{\sigma_j^f}([-1, \infty])$, respectively. We learned our final weight vector w using AdaGrad (Duchi et al., 2010), an adaptive subgradient version of standard stochastic gradient ascent.

6.2 Testing

We evaluate our model by measuring precision and recall on extracted phrase pairs. Thus, the decoding problem takes a sentence pair $(\mathbf{e}, \mathbf{f})$ as input, and must produce an extraction set π as output. Our approach, posterior thresholding, is extremely simple: we set $\pi_{ghk\ell} = \textit{true}$ iff $b_{\pi_{ghk\ell}}(\textit{true}) \geq \tau$ for some fixed threshold τ. Note that this decoding method does not require that there be any underlying word alignment a licensing the resulting extraction set π,[8] but the structure of the model is such that two conflicting phrase pairs are unlikely to simultaneously have high posterior probability.

Most publicly available translation systems expect word-level alignments as input. These can also be generated by applying posterior thresholding, aligning target word i to source word j whenever $b_{a_{ij}}(\textit{sure}) \geq t$.[9]

7 Experiments

Our experiments are performed on Chinese-to-English alignment. We trained and evaluated all models on the NIST MT02 test set, which consists of 150 training and 191 test sentences and has been used previously in alignment experiments (Ayan and Dorr, 2006; Haghighi et al., 2009; DeNero and Klein, 2010). The unsupervised HMM word aligner used to generate features for the model was trained on 11.3 million words of FBIS newswire data. We test three models: the Viterbi ITG model of DeNero and Klein (2010), our BP ITG model that uses the ITG factor, and our BP Relaxed model that replaces the ITG factor with the ONESPAN factors. In all of our experiments, the phrase length d was set to 3.[10]

7.1 Phrase Alignment

We tested the models by computing precision and recall on extracted phrase pairs, relative to the gold phrase pairs of up to length 3 induced by the gold word alignments. For the BP models, we trade off precision and recall by adjusting the decoding threshold τ. The Viterbi ITG model was trained to optimize F_5, a recall-biased measure, so in addition to F_1, we also report the recall-biased F_2 and F_5 measures. The maximum number of BP iterations was set to 5 for the BP ITG model and to 10 for the BP Relaxed model.

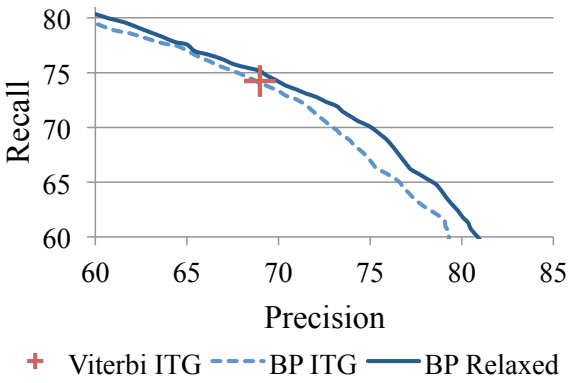
The phrase alignment results are shown in Figure 4. The BP ITG model performs comparably to the Viterbi ITG model. However, because posterior decoding permits explicit tradeoffs between precision and recall, it can do much better in the recall-biased measures, even though the Viterbi ITG model was explicitly trained to maximize F_5 (DeNero and

[7] See Section 7.2 for an empirical investigation of this maximum.

[8] This would be true even if we computed posteriors exactly, but is especially true with approximate marginals from BP, which are not necessarily consistent.

[9] For our experiments, we set $t = 0.2$.

[10] Because the runtime of the Viterbi ITG model grows exponentially with d, it was not feasible to perform comparisons for higher phrase lengths.



Model	Best Scores			Sentences
	F_1	F_2	F_5	per Second
Viterbi ITG	71.6	73.1	74.0	0.21
BP ITG	71.8	74.8	83.5	0.11
BP Relaxed	**72.6**	**75.2**	**84.5**	**1.15**

Figure 4: Phrase alignment results. A portion of the Precision/Recall curve is plotted for the BP models, with the result from the Viterbi ITG model provided for reference.

Klein, 2010). The BP Relaxed model performs the best of all, consistently achieving higher recall for fixed precision than either of the other models. Because of its lower asymptotic runtime, it is also much faster: over 5 times as fast as the Viterbi ITG model and over 10 times as fast as the BP ITG model.[11]

7.2 Timing

BP approximates marginal posteriors by iteratively updating beliefs for each variable based on current beliefs about other variables. The iterative nature of the algorithm permits us to make an explicit speed/accuracy tradeoff by limiting the number of iterations. We tested this tradeoff by limiting both of the BP models to run for 2, 3, 5, 10, and 20 iterations. The results are shown in Figure 5. Neither model benefits from running more iterations than used to obtain the results in Figure 4, but each can be sped up by a factor of almost 1.5x in exchange for a modest (< 1 F_1) drop in accuracy.

[11]The speed advantage of Viterbi ITG over BP ITG comes from Viterbi ITG's aggressive beaming.

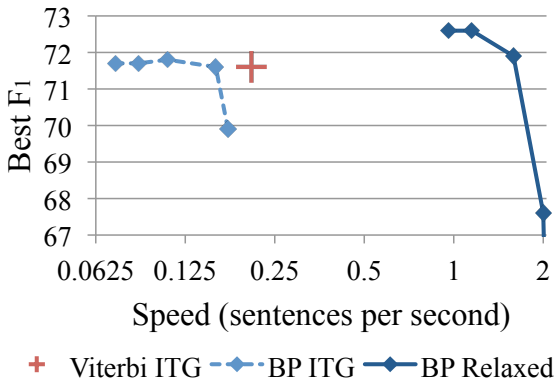


Figure 5: Speed/accuracy tradeoff. The speed axis is on a logarithmic scale. From fastest to slowest, data points correspond to maximums of 2, 5, 10, and 20 BP iterations. F_1 for the BP Relaxed model was very low when limited to 2 iterations, so that data point is outside the visible area of the graph.

Model	BLEU	Relative Improve.	Hours to Train/Align
Baseline	32.8	+0.0	5
Viterbi ITG	33.5	+0.7	831
BP Relaxed	33.6	+0.8	39

Table 2: Machine translation results.

7.3 Translation

We ran translation experiments using Moses (Koehn et al., 2007), which we trained on a 22.1 million word parallel corpus from the GALE program. We compared alignments generated by the baseline HMM model, the Viterbi ITG model and the Relaxed BP model.[12] The systems were tuned and evaluated on sentences up to length 40 from the NIST MT04 and MT05 test sets. The results, shown in Table 2, show that the BP Relaxed model achives a 0.8 BLEU improvement over the HMM baseline, comparable to that of the Viterbi ITG model, but taking a fraction of the time,[13] making the BP Relaxed model a practical alternative for real translation applications.

[12]Following a simplified version of the procedure described by DeNero and Klein (2010), we added rule counts from the HMM alignments to the extraction set aligners' counts.

[13]Some of the speed difference between the BP Relaxed and Viterbi ITG models comes from better parallelizability due to drastically reduced memory overhead of the BP Relaxed model.

8 Conclusion

For performing inference in a state-of-the-art, but inefficient, alignment model, belief propagation is a viable alternative to greedy search methods, such as beaming. BP also results in models that are much more scalable, by reducing the asymptotic complexity of inference. Perhaps most importantly, BP permits the relaxation of artificial constraints that are generally taken for granted as being necessary for efficient inference. In particular, a relatively modest relaxation of the ITG constraint can directly be applied to any model that uses ITG-based inference (e.g. Zhang and Gildea, 2005; Cherry and Lin, 2007; Haghighi et al., 2009).

Acknowledgements

This project is funded by an NSF graduate research fellowship to the first author and by BBN under DARPA contract HR0011-06-C-0022.

References

Necip Fazil Ayan and Bonnie J. Dorr. 2006. Going beyond AER: An extensive analysis of word alignments and their impact on MT. In *ACL*.

Alexandra Birch, Chris Callison-Burch, and Miles Osborne. 2006. Constraining the phrase-based, joint probability statistical translation model. In *AMTA*.

Phil Blunsom, Trevor Cohn, Chris Dyer, and Miles Osborne. 2009. A gibbs sampler for phrasal synchronous grammar induction. In *ACL-IJCNLP*.

Colin Cherry and Dekang Lin. 2007. Inversion transduction grammar for joint phrasal translation modeling. In *NAACL Workshop on Syntax and Structure in Statistical Translation*.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

Fabien Cromières and Sadao Kurohashi. 2009. An alignment algorithm using belief propagation and a structure-based distortion model. In *EACL*.

Steve DeNeefe, Kevin Knight, Wei Wang, and Daniel Marcu. 2007. What can syntax-based MT learn from phrase-based MT? In *EMNLP-CoNLL*.

John DeNero and Dan Klein. 2010. Discriminative modeling of extraction sets for machine translation. In *ACL*.

John DeNero, Dan Gillick, James Zhang, and Dan Klein. 2006. Why generative phrase models underperform surface heuristics. In *NAACL Workshop on Statistical Machine Translation*.

John DeNero, Alexandre Bouchard-Côté, and Dan Klein. 2008. Sampling alignment structure under a Bayesian translation model. In *EMNLP*.

John DeNero. 2010. Personal Communication.

John Duchi, Danny Tarlow, Gal Elidan, and Daphne Koller. 2007. Using combinatorial optimization within max-product belief propagation. In *NIPS 2006*.

John Duchi, Elad Hazan, and Yoram Singer. 2010. Adaptive subgradient methods for online learning and stochastic optimization. In *COLT*.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *HLT-NAACL*.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *COLING-ACL*.

Aria Haghighi, John Blitzer, John DeNero, and Dan Klein. 2009. Better word alignments with supervised ITG models. In *ACL-IJCNLP*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *ACL*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL*.

Zhifei Li and Sanjeev Khudanpur. 2008. A scalable decoder for parsing-based machine translation with equivalent language model state maintenance. In *ACL SSST*.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *HLT-NAACL*.

David J.C. MacKay. 2003. *Information theory, inference, and learning algorithms*. Cambridge Univ Press.

Daniel Marcu and Daniel Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *EMNLP*.

Jan Niehues and Stephan Vogel. 2008. Discriminative word alignment via alignment matrix modeling. In *ACL Workshop on Statistical Machine Translation*.

David A. Smith and Jason Eisner. 2008. Dependency parsing by belief propagation. In *EMNLP*.

Ben Taskar, Simon Lacoste-Julien, and Dan Klein. 2005. A discriminative matching approach to word alignment. In *EMNLP*.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–404.

Hao Zhang and Daniel Gildea. 2005. Stochastic lexicalized inversion transduction grammar for alignment. In *ACL*.

Hao Zhang, Chris Quirk, Robert C. Moore, and Daniel Gildea. 2008. Bayesian learning of non-compositional phrases with synchronous parsing. In *ACL:HLT*.