Monte Carlo techniques for phrase-based translation[†]

Abhishek Arun (a.arun@sms.ed.ac.uk)
Barry Haddow (bhaddow@inf.ed.ac.uk)
Philipp Koehn (pkoehn@inf.ed.ac.uk)
Adam Lopez (alopez@inf.ed.ac.uk)
University of Edinburgh

Chris Dyer (redpony@umd.edu)
University of Maryland, College Park

Phil Blunsom (pblunsom@comlab.ox.ac.uk)
Oxford University Computing Laboratory

Abstract. Recent advances in statistical machine translation have used approximate beam search for NP-complete inference within probabilistic translation models. We present an alternative approach of sampling from the posterior distribution defined by a translation model. We define a novel Gibbs sampler for sampling translations given a source sentence and show that it effectively explores this posterior distribution. In doing so we overcome the limitations of heuristic beam search and obtain theoretically sound solutions to inference problems such as finding the maximum probability translation and minimum risk training and decoding.

Keywords: Statistical Machine Translation, Gibbs sampling, Machine Learning, MCMC

1. Introduction

Statistical machine translation (SMT) poses the problem: given a foreign sentence f, find the translation e^* that maximises the posterior probability $p(e|f)$. Translation models, such as the phrase-based translation model that we focus on in this paper (Koehn et al., 2003), define multiple derivations for each translation, making the probability of a translation the sum over all of its derivations. Unfortunately, finding the maximum probability translation is NP-hard for this model (Casacuberta and Higuera, 2000), making approximations necessary. The most common of these approximations is the Viterbi approximation, which can be computed in polynomial time via dynamic programming (DP). While fast and effective for many problems, it has two serious drawbacks for probabilistic inference. First, the error incurred by the Viterbi maximum with respect to the true model maximum is unbounded. Second, the DP solution requires substantial pruning and restricts the use of non-local features. The latter problem persists even in the variational approximations of Li et al. (2009), who attempt to solve the former.

[†] This paper extends work presented in Arun et al. (2009).

We address these problems with Monte Carlo techniques. Our solution is a Gibbs sampler that draws samples from the posterior distribution of a phrase-based translation model (Section 2). Experiments reveal that our sampler effectively explores the posterior distribution (Section 3) and enables maximum probability and minimum risk decoding (Section 4). We present new results on three datasets showing that these techniques give competitive results with respect to the standard phrase-based MT pipeline (Section 5).

2. A Gibbs Sampler for Phrase based Statistical Machine Translation

A phrase-based translation model (Koehn et al., 2003) segments input sentence f of length m into phrases, which are sequences of adjacent words. Each phrase is translated into a target phrase, producing an output sentence e and an alignment a representing the mapping from source to target positions. Phrases are also reordered during translation.

We use a log-linear model on features $\mathbf{h}$, parametrised by weights θ.

$$P(e,a|f;\theta) = \frac{\exp\left[\theta \cdot \mathbf{h}(e,a,f)\right]}{\sum_{\langle e',a'\rangle} \exp\left[\theta \cdot \mathbf{h}(e',a',f)\right]} \tag{1}$$

A parameter Λ limits the number of source language words that intervene between adjacent target phrases. In our experiments, $\Lambda = 6$.

Gibbs Sampling We use Markov chain Monte Carlo (MCMC) sampling for inference in this model. MCMC probabilistically generates sample derivations from the complete search space. The probability of generating each sample is conditioned on the previous sample, forming a Markov chain. Eventually, this chain converges to the desired distribution. We use Gibbs sampling (Geman and Geman, 1984) which obtains samples from the joint distribution of a set of random variables $X = \{X_1, \ldots, X_n\}$ by sampling each variable at a time from its conditional distribution.

We require our Gibbs sampler to produce a sequence of samples, $\mathcal{S}_1^N = \{(e_i, a_i)\}_{i=1}^N$, that are drawn from the distribution $P(e,a|f)$. We can use the samples to estimate the expectation of a function $h(e,a,f)$ as follows:

$$\mathbb{E}_{P(a,e|f)}[h] = \lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} h(a_i, e_i, f) \tag{2}$$

Taking h to be an indicator function $h = \delta(a,\hat{a})\delta(e,\hat{e})$ provides an estimate of $P(\hat{a},\hat{e}|f)$, and using $h = \delta(e,\hat{e})$ marginalises over all derivations a', yielding an estimate of $P(\hat{e}|f)$.

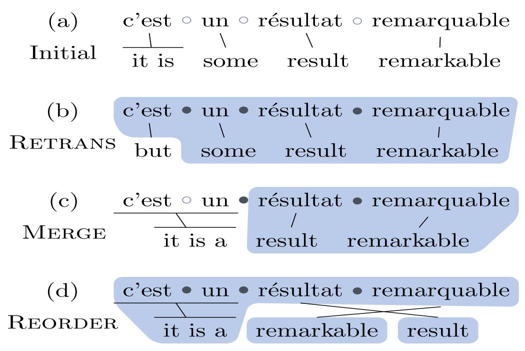
(a) Initial
c'est ○ un ○ résultat ○ remarquable
it is some result remarkable
(b) Retrans
c'est ● un ● résultat ● remarquable
but some result remarkable
(c) Merge
c'est ○ un ● résultat ● remarquable
it is a result remarkable
(d) Reorder
c'est ● un ● résultat ● remarquable
it is a remarkable result

Figure 1. Example evolution of an initial hypothesis via application of several operators. Variables that stay constant during each sampling step are indicated by shading.

2.1. Sampler Description

The sampler consists of simple operators that when concatenated enable it to efficiently explore the distribution. Each operator proposes a small change to the existing translation; the likelihood of accepting the change is proportional to its conditional probability with respect to the unchanged remainder of the translation. Given an initial sample, an iteration of the sampler consists of applying each operator at each possible point in the sentence. A new sample is then collected.

Our sampler consists of three operators. Retrans varies the translation of a single source phrase. Segmentation, alignment, and all other target phrases are held constant. Merge-Split varies the source segmentation at a single word boundary. If the boundary is a split point in the current hypothesis, the adjoining phrases can be merged, provided that the corresponding target phrases are adjacent and the phrase table contains a translation of the merged phrase. If the boundary is not a split point, the covering phrase may be split, provided that the phrase table contains a translation of both new phrases. Remaining segmentation points, phrase alignment and target phrases are held constant. Reorder varies the target phrase order for a pair of source phrases, provided that the new alignment does not violate reordering limit Λ. Segmentation, phrase translations, and all other alignments are held constant. Figure 1 illustrates sampling using the operators in our model.

The log-linear model of Equation 1 is effectively defined over the features of the phrase-pairs and alignments involved in the translation. While the Retrans and Reorder operators keep the number of phrase-pairs/alignments used in a translation constant, the Merge-Split operator can vary this number. However, Gibbs sampling is suitable only for a distribution on a fixed number of variables. If the dimensionality is variable, then we must use alternate methods such as reversible-jump Monte Carlo (Green, 1995). To show that we are actually computing a distribution on a fixed number of variables, we will use an alternate representation. We must first formally define some variables.

- Let i and j be inter-word source indices where$0 \leq i \leq j \leq m$.
- Let $[i,j]$ denote a source span. The left frontier of the span denotes position i and its right frontier refers to position j.
- A source span is *active* if $[i,j]$ is a current segmentation in source sentence f and is *inactive* otherwise.
- Let f_{ij} be the source phrase spanning i to j in sentence f.
- Let E represent a target side phrase and $\mathcal{E}$ the set of all target side phrases.
- Then, $T_{[i,j,E]}$ is an indicator variable defined as follows.

$$T_{[i,j,E]} = \begin{cases} 1 & \text{if } f_{ij} \text{ translates to E in the translation sequence } f \rightarrow (e,a) \\ 0 & \text{otherwise} \end{cases}$$

  In other words, $T_{[i,j,E]}$ denotes a phrase-pair with f_{ij} as its source and E as its target.
- Let T consist of all $T_{[i,j,E]}$ variables.
- Let $S_{[j,i]}$ be an indicator variable defined as follows.

$$S_{[j,i]} = \begin{cases} 1 & \text{if a span with right frontier } j \text{ is translated immediately before} \\ & \text{a span with left frontier } i \text{ in the translation } \mathbf{f} \rightarrow (\mathbf{e},\mathbf{d}) \\ 0 & \text{otherwise} \end{cases}$$

- Let S consist of all $S_{[j,i]}$ variables.

The $T_{[i,j,E]}$ variables represent phrase pairs involved in a translation and the $S_{[j,i]}$ variables capture the alignment sequence of these phrase pairs. We denote an indicator variable with value equal to 1 as *active* and *inactive* otherwise.

We cannot freely assign any set of values to our variables. There are several constraints. Firstly, there can only be one active phrase-pair variable per *active source span*.

$$\sum_{E \in \mathcal{E}} T_{[i,j,E]} = 1, \forall i,j : [i,j] \text{ is an active source span} \tag{3}$$

Second, only one alignment variable may be active for the right frontier of a span; likewise for the left frontier.

$$\sum_{i'} S_{[j,i']} = 1, \forall j : \text{right frontier of an active source span} \tag{4}$$

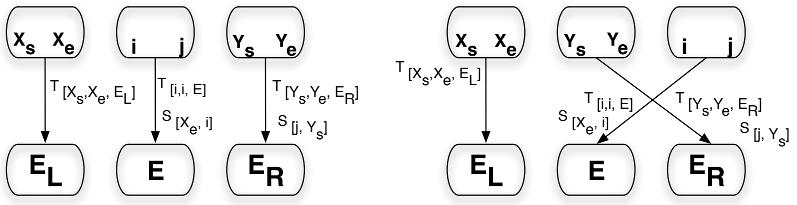

Figure 2. The left hand side diagram shows a monotone translation. Figure on the right shows a translation with source side reordering. Source phrases are annotated with their spans. Each translation step is annotated with its associated active phrase-pair and alignment variables. For example, translating source words spanning $[i,j]$ to target phrase E is captured by the phrase-pair variable $T_{[i,j,E]}$ and the alignment variable $S_{[X_e,i]}$, where X_e is the end position of the source span of the target phrase translated immediately before E.

$$\sum_{j'} S_{[j',i]} \;=\; 1, \forall i : \text{left frontier of an active source span} \tag{5}$$

Given valid configurations of T and S, we can easily reconstruct (e,a). Figure 2 gives an example of two translation hypotheses annotated with active phrase-pair and alignment variables.

Features $\mathbf{h}(e,a,f)$ in Equation 1 can be decomposed into simpler functions depending on mostly local information. Assume a phrase-based model with 4 such features:

1. A translation model feature with weight θ_T and score $\mathbf{h}_T(E, f_{ij})$.
2. A word penalty feature with weight θ_W and score $\mathbf{h}_W(E)$.
3. A linear distortion feature with weight θ_D and score $\mathbf{h}_D(j,i)$.
4. A language model (LM) feature with weight θ_L. The LM contribution of phrase-pair $T_{[i,j,E]}$, given the alignments S in the translation, is represented as the triple $\mathbf{h}_L(E, e_-^{[S,i,j]}, e_+^{[S,i,j]})$ where $e_-^{[S,i,j]}$ encodes the LM pre-context of $T_{[i,j,E]}$ and $e_+^{[S,i,j]}$ its LM post-context.

The model in Equation 1 can now be factorised as:

$$\begin{aligned} P(e,a|f;\theta) &= P(T,S|f;\theta) \\ &\propto \exp\left[\theta \cdot \mathbf{h}(e,a,f)\right] \\ &= \exp\ \theta_T \sum_{T_{[i,j,E]}\in T} \left[T_{[i,j,E]}\mathbf{h}_T(E,f_{ij})\right] \cdot \exp\ \theta_W \sum_{T_{[i,j,E]}\in T} \left[T_{[i,j,E]}\mathbf{h}_W(E)\right] \cdot \\ &\quad \exp\ \theta_D \sum_{S_{[j,i]}\in S} \left[S_{[j,i]}\mathbf{h}_D(j,i)\right] \cdot \\ &\quad \exp\ \theta_L \sum_{T_{[i,j,E]}\in T} \left[T_{[i,j,E]}\mathbf{h}_L(E, e_-^{[S,i,j]}, e_+^{[S,i,j]})\right] \end{aligned} \tag{6}$$

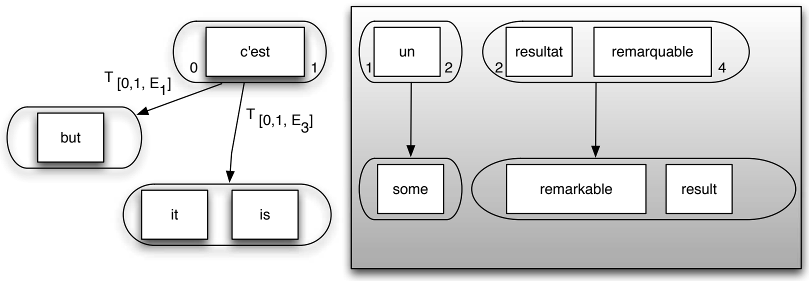

Figure 3. Example of a RETRANS step for block $T_{[0,1]} = \{T_{[0,1,E_1]}, T_{[0,1,E_3]}\}$ where E_1 = “but" and E_3 = “it is". The variables $T_{[0,1,E_1]}$ and $T_{[0,1,E_3]}$ correspond to the phrase-pair ⟨“c’est", “but"⟩ and ⟨“c’est", “it is"⟩ respectively. Source phrases are annotated with their spans. The shaded box covers all variables that stay constant during the sampling step. All alignment variables stay fixed.

Since the model is defined over fixed-length T and S, we can apply Gibbs sampling to it. In basic Gibbs sampling we would deterministically scan the variables left-to-right, resampling each in turn. However, due to the deterministic constraints between variables, we use a *blocked sampling* strategy whereby mutually constrained variables are sampled together. To do this we define blocks of variables that allow us to vary their assignments while respecting the constraints in Equations 3, 4 and 5 respectively:

- Let $T_{[i,j]}$ be the set of all phrase-pair variables spanning $[i, j]$.
- Let $S_{[j,-]} = \{S_{[j',i']} | j' = j\}$ be the set of all alignment variables such that j is the right frontier of a source phrase translated immediately before another phrase.
- Let $S_{[-,i]} = \{S_{[j',i']} | i' = i\}$ be the set of all alignment variables such that i is the left frontier of a source phrase translated immediately after another phrase.

We are now in a position to formally describe the operators.

3. Sampler Operators

3.1. RETRANS

Given a source span $[i, j]$, RETRANS chooses a new target phrase from the block $T_{[i,j]}$. Assuming we want to sample from $T_{[I,J]} = \{T_{[I,J,E_1]}, T_{[I,J,E_2]} \cdots T_{[I,J,E_n]}\}$. The conditional probability of sampling $T_{[I,J,E_e]}$ $(1 \leq e \leq n)$ is given by:

$$P(T_{[I,J,E_e]} \mid T_{[I\backslash,J\backslash]}, S) = \frac{P(T_{[I,J,E_e]}, T_{[I\backslash,J\backslash]}, S)}{\sum_{i=1}^{n} P(T_{[I,J,E_i]}, T_{[I\backslash,J\backslash]}, S)}$$

where $T_{[i\backslash,j\backslash]} = \left[T_{[i',j',E]} \mid i' \neq i \text{ and } j' \neq j\right]$ is a vector consisting of phrase-pair variables that do not span $[i, j]$.

This implies that we would compute the score of entire translations, which is clearly expensive. However, we can factorise the joint distribution (Equation 6) as a product of variables resampled by RETRANS ($T_{[I,J,E_e]}$) and constant variables. The constant terms cancel so that the conditionals reduce to the form:

$$P(T_{[I,J,E_e]} \mid T_{[I\backslash,J\backslash]}, S) = \frac{\exp\left[\theta_T \mathbf{h}_T(E_e, f_{IJ}) + \theta_L \mathbf{h}_L(E_e, e_-^{[S,I,J]}, e_+^{[S,I,J]}) + \theta_W \mathbf{h}_W(E_e)\right]}{\sum_{i=1}^{n} \exp\left[\theta_T \mathbf{h}_T(E_i, f_{IJ}) + \theta_L \mathbf{h}_L(E_i, e_-^{[S,I,J]}, e_+^{[S,I,J]}) + \theta_W \mathbf{h}_W(E_i)\right]} \quad (7)$$

RETRANS proposals are thus proportional to the scores of the phrases in the block, which can be computed efficiently.

Example Figure 3 shows an example of the RETRANS operator. We want to sample from $T_{[0,1]} = \{T_{[0,1,E_1]}, T_{[0,1,E_3]}\}$ where $f_{0,1} =$ "*c'est*", $E_1 =$ "*but*" and $E_3 =$ "*it is*". Denote the start of sentence marker with $\langle s \rangle$ and set all feature weights to 1.
Then,

$$\begin{aligned}
P(T_{[0,1,E_1]} \mid T_{[I\backslash,J\backslash]}, S) &= \frac{\exp\left[\mathbf{h}_T(\text{“but”}, \text{“c'est”}) + \mathbf{h}_L(\text{“but”}, \langle s \rangle, \text{“some”}) + \mathbf{h}_W(\text{“but”})\right]}{Z} \\
P(T_{[0,1,E_3]} \mid T_{[I\backslash,J\backslash]}, S) &= \frac{\exp\left[\mathbf{h}_T(\text{“it is”}, \text{“c'est”}) + \mathbf{h}_L(\text{“it is”}, \langle s \rangle, \text{“some”}) + \mathbf{h}_W(\text{“it is”})\right]}{Z} \\
\text{where } Z &= \exp\left[\mathbf{h}_T(\text{“but”}, \text{“c'est”}) + \mathbf{h}_L(\text{“but”}, \langle s \rangle, \text{“some”}) + \mathbf{h}_W(\text{“but”})\right] \\
&+ \exp\left[\mathbf{h}_T(\text{“it is”}, \text{“c'est”}) + \mathbf{h}_L(\text{“it is”}, \langle s \rangle, \text{“some”}) + \mathbf{h}_W(\text{“it is”})\right]
\end{aligned}$$

RETRANS then resamples a target phrase E from this conditional distribution.

3.2. REORDER

This operator takes a pair of source spans $[i, j]$ and $[k, l]$ and samples new values for the alignment variables from the blocks $S_{[-,i]}$, $S_{[-,k]}$, $S_{[j,-]}$ and $S_{[l,-]}$, such that reordering limit constraints are respected. There are two possible outcomes to each REORDER operation: maintain the current alignments or swap the alignments.

Assume current active alignments $S_{[x_1,i]}$, $S_{[j,x_2]}$, $S_{[x_3,k]}$, and $S_{[l,x_4]}$ and proposed swapped alignments $S_{[x_3,i]}, S_{[j,x_4]}, S_{[x_1,k]}$ and $S_{[l,x_2]}$. The required conditional probabilities are:

$$\begin{aligned}
P(S_{[x_1,i]}, S_{[j,x_2]}, S_{[x_3,k]}, S_{[l,x_4]} \mid S\backslash, T) &= (P(S_{[x_1,i]}, S_{[j,x_2]}, S_{[x_3,k]}, S_{[l,x_4]}, S\backslash, T))/Z \\
P(S_{[x_3,i]}, S_{[j,x_4]}, S_{[x_1,k]}, S_{[l,x_2]} \mid S\backslash, T) &= (P(S_{[x_3,i]}, S_{[j,x_4]}, S_{[x_1,k]}, S_{[l,x_2]}, S\backslash, T))/Z
\end{aligned}$$

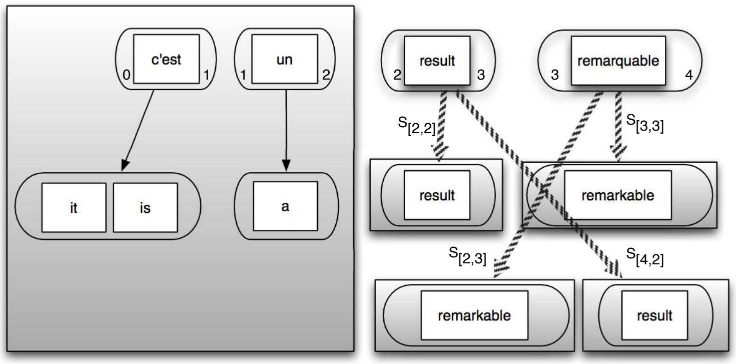

Figure 4. Example of a REORDER step for source spans [2,3] and [3,4]. The operator considers a monotone alignment (activating $S_{[2,2]}$ and $S_{[3,3]}$) and a reordered alignment (activating $S_{[2,3]}$ and $S_{[4,2]}$). Source phrases are annotated with their spans. Shaded boxes cover all variables that stay constant during the sampling step. All phrase-pair variables stay fixed.

where

$$\begin{aligned} S\backslash &= \{S_{[j',i']} | (j',i') \not\in \{(x_1,i),(x_3,i),(j,x_2),(j,x_4),(x_3,k),(x_1,k),(l,x_4),(l,x_2)\} \text{ and} \\ Z &= P(S_{[x_1,i]},S_{[j,x_2]},S_{[x_3,k]},S_{[l,x_4]},S\backslash,T) + P(S_{[x_3,i]},S_{[j,x_4]},S_{[x_1,k]},S_{[l,x_2]},S\backslash,T) \end{aligned}$$

As with RETRANS, we can factor out constant terms. These are word penalty and translation model scores for all phrase pairs and distortion and language model scores for all alignment blocks that are held constant. For each of the two alignment possibilities, the conditional probabilities reduce to calculating 4 distortion scores and 2 language model scores. Note however that if the alignments are adjacent on both source and target side and translated monotonically with respect to each other, then only 3 distortion scores need to be computed.

Example We illustrate this operator using the example in Figure 4 in which the sampler considers reordering the alignments at source spans $[2,3]$ and $[3,4]$. There are 2 possible outcomes to each reorder operation : (a) maintain the current alignment or (b) swap the alignment (since doing so does not violate reordering constraints).

The blocks being sampled from are $S_{[-,2]}$, $S_{[-,3]}$, $S_{[3,-]}$ and $S_{[4,-]}$. The monotone alignment is represented by $S_{[2,2]}$, $S_{[3,3]}$ (duplicated as the phrases are adjacent on the target side) and $S_{[4,\langle \backslash s \rangle]}$ where $\langle \backslash s \rangle$ denotes the end of sentence marker . By definition $S_{4,\langle \backslash s \rangle}$ has a score of 0 so we eliminate the term from our calculations. Also, we remove the duplicate alignment variable leaving us with $S_{[2,2]}$ and $S_{[3,3]}$

The swapped alignment is represented by $S_{[4,2]}$, $S_{[2,3]}$, $S_{[3,\langle\backslash s\rangle]}$ and $S_{[4,2]}$. Removing the duplicate variable and the variable involving $\langle\backslash s\rangle$ leaves us with $S_{[4,2]}$ and $S_{[2,3]}$.

Assuming

$$S\backslash \;=\; \{S_{[j',i']} \,|\, [j',i'] \not\in \{[2,2]\,,[2,3]\,,[3,3]\,,[4,2]\}\}$$

The conditional probabilities for sampling are:

$$\begin{aligned}
P(S_{[2,2]}, S_{[3,3]}|S\backslash, T) &= \frac{P(S_{[2,2]}, S_{[3,3]}, S\backslash, T)}{P(S_{[2,2]}, S_{[3,3]}, S\backslash, T) + P(S_{[4,2]}, S_{[2,3]}, S\backslash, T)} \\
&= \frac{\exp\left[\mathbf{h}_D(2,2) + \mathbf{h}_D(3,3) + \mathbf{h}_L(\text{“result remarkable"}, \text{“a"}, \langle\text{\textbackslash s}\rangle)\right]}{Z} \\
P(S_{[4,2]}, S_{[2,3]}|S\backslash, T) &= \frac{P(S_{[4,2]}, S_{[2,3]}, S\backslash, T)}{P(S_{[2,2]}, S_{[3,3]}, S\backslash, T) + P(S_{[4,2]}, S_{[2,3]}, S\backslash, T)} \\
&= \frac{\exp\left[\mathbf{h}_D(4,2) + \mathbf{h}_D(2,3) + \mathbf{h}_L(\text{“remarkable result"}, \text{“a"}, \langle\text{\textbackslash s}\rangle)\right]}{Z} \\
\text{where } Z &= (\exp\left[\mathbf{h}_D(2,2) + \mathbf{h}_D(3,3) + \mathbf{h}_L(\text{“result remarkable"}, \text{“a"}, \langle\text{\textbackslash s}\rangle)\right]) \\
&+ (\exp\left[\mathbf{h}_D(4,2) + \mathbf{h}_D(2,3) + \mathbf{h}_L(\text{“remarkable result"}, \text{“a"}, \langle\text{\textbackslash s}\rangle)\right])
\end{aligned}$$

3.3. MERGE-SPLIT

The first 2 operators considered so far keep the number of source side segments and therefore the number of active phrase-pairs/alignments in the model constant. The MERGE-SPLIT operator, on the other hand, looks to increase this number (by performing a split operation) or decrease this number (by merging) or keep it constant.

MERGE: Given a position j such that $[i, j]$ and $[j,k]$ are active spans, the MERGE operator samples from all the possible ways of translating the span $[i,k]$, either by maintaining the current segmentations or by merging the segmentations in to one span. Reordering is not allowed during this sampling operation.

The operator first considers all the possibilities of translating $[i,k]$ using the variables in the blocks $T_{[i,j]}$ and $T_{[j,k]}$. Additionally, if existing spans $[i, j]$ and $[j,k]$ are currently being translated monotonically with respect to each other and if their translations are adjacent on the target side, i.e. $S_{[j,j]}$ = 1, then the operator also considers variables from the block $T_{[i,k]}$. The operator then samples a new configuration for the variables.

If the operator chooses to merge the segmentations, it has to:

- activate the new segmentation $[i,k]$ by activating one variable from the $T_{[i,k]}$ block.

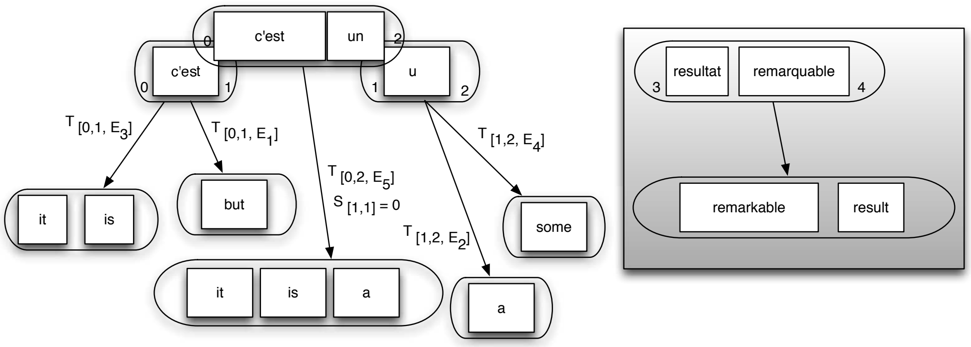

Figure 5. Example of a MERGE involving source spans [0,1], [0,2] and [1,2]. The operator considers translating the source span [0,2] using one phrase-pair or by maintaining the current segmentations. Here, E_1 = “but", E_2 = “a", E_3 = “it is", E_4 = “some" and E_5 = “it is a". Merging span [0,2] by activating $T_{[0,2,E_5]}$ requires setting off the alignment variable $S_{[1,1]}$. The shaded box covers variables that stay constant during the sampling step.

- inactivate the segmentations $[i, j]$ and $[j,k]$ by turning off all variables in $T_{[i,j]}$ and $T_{[j,k]}$ and by setting $S_{[j,j]}$ to 0.

The case where the operator chooses to maintain the current segmentations is equivalent to performing the RETRANS operator on each of the blocks $T_{[i,j]}$ and $T_{[j,k]}$.

Figure 5 illustrates the MERGE operator. The span [0,2] can be either translated by sampling from the block $T_{[0,2]} = \{T_{[0,2,E_5]}\}$ or by maintaining the current segmentations and sampling from blocks $T_{[0,1]} = \{T_{[0,1,E_1]}, T_{[0,1,E_3]}\}$ and $T_{[1,2]} = \{T_{[1,2,E_2]}, T_{[1,2,E_4]}\}$. In the latter case, the operator considers the set of variables formed by a cartesian product over the two blocks. In total, the operator considers 5 possible phrase-pair variable assignment configurations.

SPLIT: The split operator is the converse of the MERGE operator. Given a position j ($i < j < k$) such that the block $T_{[i,k]}$ has an active phrase-pair variable, the split operator samples from the phrase-pair blocks $T_{[i,j]}$, $T_{[j,k]}$ and $T_{[i,k]}$. Reordering is not allowed during this sampling operation.

If the operator decides to split the current segmentation, then it has to:

- activate one variable from each of the $T_{[i,j]}$ and $T_{[j,k]}$ blocks and turn off all variables in the $T_{[i,k]}$ block.
- set the value of the alignment variable $S_{[j,j]}$ to 1.

In case the operator decides against splitting, it samples a new phrase-pair assignment from the block $T_{[i,k]}$ (this is equivalent to a RETRANS operation).

The MERGE-SPLIT operator can therefore be seen as trying to translate a source span $[i,k]$ either with one phrase-pair or with two source adjacent phrase-pairs while leaving distortions constant. Conditional probabilities are derived in a manner similar to those for RETRANS.

3.4. ALGORITHMIC COMPLEXITY

Since both the RETRANS and MERGE-SPLIT operators are applied by iterating over source side word positions, their complexity is linear in n, the size of the input. The REORDER operator iterates over the positions in the input and for the source phrase found at that position considers swapping its target phrase with that of every other source phrase, *provided* that the reordering limit is not violated. This means that it can only consider swaps within a fixed-length window, so complexity is linear in sentence length.

The complexity of the RETRANS and MERGE-SPLIT operators also depends on the number of target phrases that have to be considered for each source phrase. Typically, only the p most probable such target phrases are retained in the model. The complexity of the RETRANS operator is therefore $O(np)$ and, since it operates over pairs of source spans, MERGE-SPLIT's complexity is $O(np^2)$. In the experiments in this work, we set p to 20.

3.5. EXPERIMENTAL VERIFICATION

To verify that our sampler was behaving as expected, we computed the KL divergence between its inferred distribution $\hat{q}(e|f)$ and the true distribution over a single sentence (Figure 6). We computed the true posterior distribution $p(e|f)$ under an Arabic-English phrase-based translation model with parameters trained to maximise expected BLEU (Section 5), summing out the derivations for identical translations and computing the partition term $Z(f)$. As the number of iterations increases, the KL divergence between the distributions approaches zero, indicating that the sampler is able to approximate the true distribution effectively.

4. Decoding

Decoding requires a search for the translation e^* that maximises or minimises some criterion given a source sentence f. We consider three common approaches to decoding, maximum translation (MaxTrans), maximum derivation (MaxDeriv), and minimum Bayes risk decoding (MBR):

$$e^* = \begin{cases} \arg\max_{(e,a)} p(e,a|f) & \text{(MaxDeriv)} \\ \arg\max_e p(e|f) & \text{(MaxTrans)} \\ \arg\min_e \sum_{e'} \ell_{e'}(e) p(e'|f) & \text{(MBR)} \end{cases} \tag{8}$$

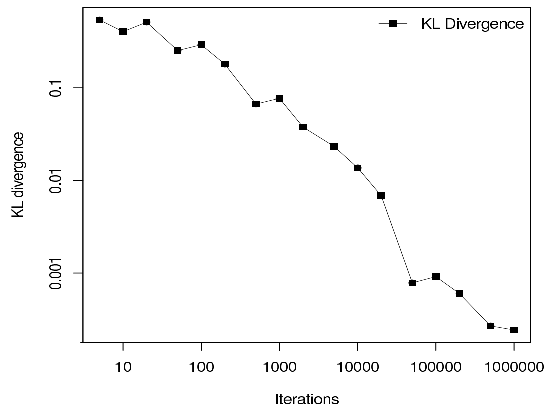

Figure 6. The KL divergence of the true posterior distribution and the distribution estimated by the Gibbs sampler at different numbers of iterations for the Arabic source sentence *r}ys wzrA' mAlyzyA yzwr Alflbyn* (in English, *The prime minister of Malaysia visits the Philippines*).

As noted in section 2, the Gibbs sampler can be used to provide an estimate of the probability distribution $P(a,e|f)$ and therefore determine the maximum of this distribution, in other words the most likely derivation. Furthermore, we can marginalise over alignments to estimate $P(e|f)$ and obtain the most likely translation. Our sampler can therefore be used as a decoder, either running in MaxTrans or MaxDeriv mode. Using it in this way makes max-translation decoding tractable, and so will help determine whether max-translation offers any benefit over the usual max-derivation. It also allows us to verify that it is producing valid samples from the desired distribution.

Additionally the sampler can be used for MBR decoding, a consensus decoding algorithm introduced by (Kumar and Byrne, 2004). In their work, as an approximation of the model probability distribution, the expected loss of the decoder is calculated by summing over an *n*-best list. With the Gibbs sampler, however, we should be able to obtain a much more accurate view of the model probability distribution. In the MBR decoder, $\ell_{e'}(e)$ is any real-valued loss (error) function that computes the error of one hypothesis e with respect to some reference e'. Our loss is a sentence-level approximation of $(1-\text{BLEU})$ whereby the BLEU n-gram precision counts are smoothed by adding 0.01 for n > 1.

Experimental setup We conducted experiments with the French-English and German-English corpora from the WMT09 shared translation task (Callison-Burch et al., 2009), and 300k parallel Arabic-English sentences from the NIST MT evaluation training data. The Arabic-English training data consists of the eTIRR corpus (LDC2004E72), the Arabic news corpus (LDC2004T17), the Ummah corpus (LDC2004T18), and the sentences with confidence $c > 0.995$ in the ISI automatically extracted web parallel corpus (LDC2006T02).

For all language pairs, we constructed a phrase-based translation model as described in Koehn et al. (2003), limiting the phrase length to 5. The target side of the parallel corpus was used to train a 3-gram language model. For the German and French systems, the DEV2006 set was used for model tuning and the TEST2007 (in-domain) and NEWS-DEV2009B (out-of-domain) sets for testing. For the Arabic system, the MT02 set (10 reference translations) was used for tuning and MT03 (4 reference translations) was used for evaluation. To reduce the size of the phrase table, we used the association-score technique suggested by Johnson et al. (2007a). Translation quality is reported using case-insensitive BLEU (Papineni et al., 2002).

We used Moses (Koehn and Hoang, 2007) as our baseline phrase-based beam-search decoder. Both the sampler and Moses used the same feature sets: forward and backward phrase translation log-probabilities, forward and backward lexical translation log-probabilities, phrase penalty, language model log-probability, word penalty and a linear distortion penalty.

For the experiments reported here, we used feature weights trained with minimum error rate training (MERT) (Och, 2003). Since MERT ignores the denominator in Equation 1, it is invariant with respect to the scale of the weight vector θ – the Moses implementation simply normalises the weight vector it finds by its ℓ_1-norm. However, when we use these weights in a true probabilistic model, the scaling factor affects the behaviour of the model since it determines how peaked or flat the distribution is. If the scaling factor is too small, then the distribution is too flat and the sampler spends too much time exploring unimportant probability regions. If it is too large, then the distribution is too peaked and the sampler may concentrate on a very narrow probability region. We optimised the scaling factor with respect to resulting BLEU scores when decoding a 200-sentence portion of the tuning set, finding that a coefficient of 10 worked best for fr-en and a coefficient of 6 for de-en.

The first experiment shows the effect of different initialisations and numbers of sampler iterations on max-derivation decoding performance of the sampler. We used Moses to generate the starting hypothesis, either in full DP max-derivation mode, or alternatively with restrictions on the features and reordering, or with zero weights to simulate a random initialisation, and the number of iterations varied from 100 to 200,000, with a 100 iteration burn-in in each case. For each of the test sets, we track the model score of the most frequent derivation for each input sentence and report the mean across all the sentences. Figure 7 shows the variation of the mean model score with sampler iteration, for the different starting points, and for both language pairs.

We see that the initialization had little effect on the model score of the best derivation, except with low numbers of iterations. This indicates that the sampler has good mobility around the distribution. We found that around 50,000 sampling iterations were required for fr-en and 100,000 for de-en for the sampler to give equivalent model scores to Moses, even when the model is

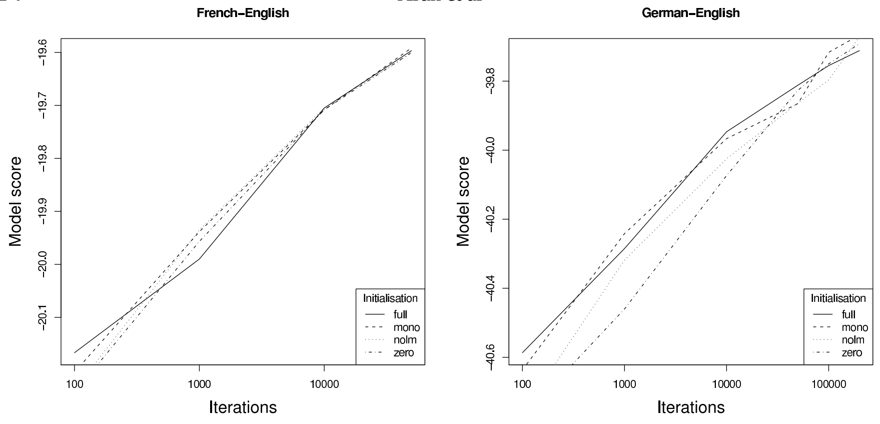

Figure 7. Mean maximum model score, as a function of iteration number and starting point. The starting point can either be the full max-derivation translation (**full**), the monotone translation (**mono**), the monotone translation with no language model (**nolm**) or the monotone translation with all weights set to zero (**zero**).

initialized with Moses run in full DP max-derivation mode. This suggests that the mobility of the sampler is such that, initially, it can wander off towards other modes of the distribution.

We are mostly interested in using the sampler to perform, at test time, more accurate MBR decoding and, at training time, better minimum risk training. Both objective functions involve computing expectations over the distribution, therefore, the ability of the sampler to explore the distribution is a desirable property. If the intent is to use the sampler for max-derivation decoding, then methods such as *simulated annealing* can be employed to accelerate convergence towards the mode of the distribution.

Speed Running the sampler for the 100,000 iterations needed to match Moses mean model scores took on average 1670 seconds per sentence on French-English and 1552 seconds per sentence on German-English. These times are prohibitively high. However, we found during the subsequent decoding experiments in this paper that running the sampler for as few as 10000 iterations gave comparable BLEU scores and that for training 4000 samples were enough to provide a good estimate of the gradient of the objective function.

Nevertheless, we do intend in future work to look at ways of speeding up the sampler. Currently, we use a *sequential scan* sampling procedure whereby a left to right scan of the input is made, applying each operator at every possible source position. To reduce correlations between successive samples, a sample is only collected at the end of the scan. This procedure is wasteful since it discards a large number of intermediate samples.

A possible way of speeding up sampling is to use a *random scan* procedure in which we first sample an operator and a source position to sample at before

Table I. Comparison of the BLEU score of the Moses decoder with the sampler running in differing decoding modes. The test sets are TEST2007 (in) and NEWS-DEV2009B (out)

	fr-en		de-en	
	in	out	in	out
Moses Max Derivation	32.7	19.1	27.4	15.9
Gibbs Max Derivation	32.6	19.1	27.0	15.5
Gibbs Max Translation	32.6	19.1	27.4	16.0
Gibbs MBR	32.6	19.2	27.3	16.0

generating the sample. Since the operator and source position are chosen at random, the correlation between successive samples is inherently reduced. To further minimize correlation, we can choose to retain only every *s*-th sample. By setting *s* to a value lower than the average lag between sample collection in sequential scanning, sampling can be speeded up by a linear factor.

Decoding algorithms In order to compare max-translation, max-derivation and MBR decoding with the Gibbs sampler and the Moses baseline, we ran experiments on both European language pairs using both in- and out-of-domain test sets. The sampler was initialized with the output of Moses with the feature weights set to zero and restricted to monotone, and run for 10,000 iterations with a 100 iteration burn-in. The scale factors were set to the same values as in the previous experiment. Relative translation quality (measured according to BLEU) is shown in Table I.

These results show very little difference between the decoding methods, indicating that the Gibbs sampling decoder can perform as well as a standard DP based max-derivation decoder with these models, and that there is no gain from doing max-translation or MBR decoding. However it should be noted that the model used for these experiments was optimised by MERT, for max-derivation decoding, and so the experiments do not rule out the possibility that max-translation and MBR decoding will offer an advantage on an appropriately optimised model.

5. Minimum risk training

In the previous section, we described how our sampler can be used to search for the best translation under a variety of decoding criteria (max derivation, translation, and minimum risk). However, there appeared to be little benefit to marginalizing over the latent derivations. This is almost certainly a side effect of the MERT training approach that was used to construct the models so as to maximise the performance of the model on its single best derivation, without regard to the shape of the rest of the distribution (Blunsom et al., 2008). In this section we describe a further application of the Gibbs sampler: to do *unbiased* minimum risk training.

While there have been at least two previous attempts to do minimum risk training for MT, both approaches relied on biased k-best approximations (Smith and Eisner, 2006; Zens et al., 2007). Since we sample from the whole distribution, we will have a more accurate risk assessment.

The risk, or expected loss, of a probabilistic translation model on a corpus $\mathcal{D}$, defined with respect to a particular loss function $\ell_{\hat{e}}(e)$, where $\hat{e}$ is the reference translation and e is a hypothesis translation, is given by:

$$\mathcal{L} = \sum_{\langle \hat{e}, f \rangle \in \mathcal{D}} \sum_{e} p(e|f) \ell_{\hat{e}}(e) \tag{9}$$

This value can be approximated using equation (2). In this section, we are concerned with finding the parameters θ that minimise (9). Fortunately, with the log-linear parameterization of $p(e|f)$, $\mathcal{L}$ is differentiable with respect to θ:

$$\frac{\partial \mathcal{L}}{\partial \theta_k} = \sum_{\langle \hat{e}, f \rangle \in \mathcal{D}} \sum_{e} p(e|f) \ell_{\hat{e}}(e) \left(h_k - \mathbb{E}_{p(e|f)}[h_k] \right) \tag{10}$$

We have now shown how to compute our objective (9), the expected loss, and a gradient with respect to the model parameters we want to optimise, (10), so we can use any standard first-order optimization technique. Since the sampler introduces stochasticity into the gradient and objective, we use stochastic gradient descent methods which are more robust to noise than more sophisticated quasi-Newton methods like L-BFGS (Liu and Nocedal, 1989). For the experiments below, we updated the learning rate after each step proportionally to difference in successive gradients (Schraudolph, 1999).

For the experiments reported in this section, we used sample sizes of 4000 and estimated the gradient on sets of 120 sentences drawn randomly (with replacement) from the development corpus. For a loss function, we use the same sentence level BLEU approximation as used for MBR decoding (Section 4). By examining performance on held-out data, we find the model converges typically in fewer than 50 iterations.

5.1. Training experiments

Table II compares the performance of translation systems tuned with MERT (maximizing corpus BLEU) with systems tuned to maximise expected sentence-level BLEU. For Arabic to English, we trained both systems with MT02 and tested on MT05, using MT03 as held-out set for the Min Risk training. For German and French to English, we trained on DEV 2006 and evaluated performance on the first 1000 sentences of WMT08 using WMT07 as held-out set. For all experiments, we used the same eight features as in Section 4. For

Table II. Decoding with minimum risk trained systems, compared with decoding with MERT–trained systems on Arabic to English MT05, German to English WMT 08 and French to English WMT 08 data. BLEU brevity penalties are indicated in parentheses. U Moses refers to Moses with phrase tables of max phrase lengths 7 and with no association score filtering. All other phrase tables have a max phrase length of 5 and are pruned using association score filtering. Best results for each language pair are indicated in bold.

Training	Decoder	AR-EN	FR-EN	DE-EN
MERT	U Moses Max Derivation	48.3 (0.995)	32.9 (1.000)	**27.9** (0.998)
	U Moses MBR	**48.4** (0.996)	32.8 (1.040)	27.8 (1.000)
MERT	Moses Max Derivation	44.3 (0.973)	33.2 (0.996)	27.8 (1.000)
	Moses MBR	44.4 (0.989)	**33.4** (1.000)	27.7 (1.000)
	Gibbs Max Derivation	43.4 (0.979)	32.4 (0.976)	26.7 (0.937)
MinRisk	Gibbs MaxTrans	43.4 (0.981)	32.7 (0.979)	26.9 (0.937)
	Gibbs MBR	43.6 (0.981)	32.8 (0.979)	27.0 (0.941)

Moses MBR experiments, we used n-best lists of 1000 distinct derivations to compute the expected loss. For all Gibbs based decoding experiments, we used 10000 samples. We also run experiments where we extracted phrase tables using the Moses default setting of max phrase length 7 (rather than 5 in our other experiments) and with no association-score pruning.

Table II confirms that on Europarl data, as shown in Johnson et al. (2007a), translating with pruned phrase tables gives similar and sometimes better translation results than with unpruned phrased tables. However, in Arabic-English, where the training set consisting of only 300K sentence pairs is small to begin with, association score pruning hurts performance substantially (-4 BLEU). For the minimum risk trained systems, we note that the positive effect of marginalizing over derivations as well as of using minimum risk decoding. Compared to MERT tuned systems, the minimum risk ones produce systematically shorter output, thus getting heavily hit by BLEU's brevity penalty. We believe that this is a result of optimizing sentence level BLEU rather than corpus BLEU as done by MERT. As future work, we intend to look at ways of directly optimizing corpus BLEU.

6. Sampler Defect

As we have shown, when the sampler is trained to optimise an appropriate objective function, translation performance is comparable to state of the art phrase-based systems. This competitive performance is achieved *in spite of* a defect of the sampler that we discovered subsequent to our previous work (Arun et al., 2009). Recall that the operators are designed to explore probability distribution in its entirety. However, suppose that a three-word phrase has a translation in our phrase table, neither of the two-word source phrases contained within it have translations, and all single words have translations. There is no sequence of our operators that can move between the three-word source segment and the single-word segments. Situations such as this can arise from phrase extraction heuristics and possible filtering of the phrase-table prior to decoding. A scan of our phrase tables shows that a few such cases exist. In future work, we intend to address this problem.

7. Related Work

Our sampler is similar to the decoder of (Germann et al., 2001; Eisner and Tromble, 2006; Langlais et al., 2007), which start with an approximate solution and then incrementally improve it via operators such as RETRANS and MERGE-SPLIT. It is also similar to the estimator of Marcu and Wong (2002), who employ the same operators to search the alignment space from a heuristic initialisation. These previous efforts employed their operators in a greedy hill-climbing search. In contrast, our operators are applied probabilistically, making them theoretically well-founded for a variety of inference problems.

Our use of Gibbs sampling follows from its increasing use in Bayesian inference problems in NLP (Finkel et al., 2006). Most closely related is the work of DeNero et al. (2008), who derive a Gibbs sampler for phrase-based alignment, using it to infer phrase translation probabilities.

To our knowledge, we are the first to apply Monte Carlo methods to maximum translation and minimum risk translation. Approaches to the former (Blunsom et al., 2008) rely on dynamic programming techniques which do not scale well without heuristic approximations, while approaches to the latter (Smith and Eisner, 2006; Zens et al., 2007) use biased *k*-best approximations.

8. Conclusion

Although the training approach presented in Section 5 has a number of theoretical advantages, its performance is in general inferior when compared with a system trained to optimise corpus level BLEU. In future work, we will explore possibilities for directly optimising BLEU.

Using sampling for model training has two further advantages that we intend to explore. First, although MERT performs quite well on models with small numbers of features (such as those we considered in this paper), in general the algorithm severely limits the number of features that can be used since it does not use gradient-based updates during optimization, instead updating one feature at a time. Our training method (Section 5) does not have this limitation, so it can use many more features.

Finally, for the DP-based max-derivation approximation to be computationally efficient, the features characterizing the steps in the derivation must be either computable independently of each other or with only limited local context (as in the case of the language model or distortion costs). This has led to a situation where entire classes of potentially useful features are not considered because they would be impractical to integrate into a DP based translation system. With the sampler this restriction is mitigated: any function of $h(e, f, a)$ may participate in the translation model subject only to its own computability.

Acknowledgments

This research was supported in part by the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-2-001; and by the EuroMatrix project funded by the European Commission (6th Framework Programme). The project made use of the resources provided by the Edinburgh Compute and Data Facility (`http://www.ecdf.ed.ac.uk/`). The ECDF is partially supported by the eDIKT initiative (`http://www.edikt.org.uk/`). We would also like to thank our two anonymous reviewers for their very detailed and insightful feedback.

References

Arun, A., C. Dyer, B. Haddow, P. Blunsom, A. Lopez, and P. Koehn: 2009, ‘Monte Carlo inference and maximization for phrase-based translation’. In: *Proceedings of CoNLL*. Boulder, Colorado, pp. 102–110.

Blunsom, P., T. Cohn, and M. Osborne: 2008, ‘A Discriminative Latent Variable Model for Statistical Machine Translation’. In: *Proceedings of ACL-08: HLT*. Columbus, Ohio, pp. 200–208.

Callison-Burch, C., P. Koehn, C. Monz, and J. Schroeder: 2009, ‘Findings of the 2009 Workshop on Statistical Machine Translation’. In: *Proceedings of the Fourth Workshop on Statistical Machine Translation*. Athens, Greece, pp. 1–28, Association for Computational Linguistics.

Casacuberta, F. and C. D. L. Higuera: 2000, *Computational Complexity of Problems on Probabilistic Grammars and Transducers*. Springer-Verlag.

DeNero, J., A. Bouchard-Côté, and D. Klein: 2008, ‘Sampling Alignment Structure under a Bayesian Translation Model’. In: *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. Honolulu, Hawaii, pp. 314–323.

Eisner, J. and R. W. Tromble: 2006, ‘Local Search with Very Large-Scale Neighborhoods for Optimal Permutations in Machine Translation’. In: *Proceedings of the HLT-NAACL Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*. New York, pp. 57–75.

Finkel, J. R., C. D. Manning, and A. Y. Ng: 2006, ‘Solving the Problem of Cascading Errors: Approximate Bayesian Inference for Linguistic Annotation Pipelines’. In: *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Sydney, Australia, pp. 618–626.

Geman, S. and D. Geman: 1984, ‘Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images’. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6**, 721–741.

Germann, U., M. Jahr, K. Knight, D. Marcu, and K. Yamada: 2001, 'Fast Decoding and Optimal Decoding for Machine Translation'. In: *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*. Toulouse, France, pp. 228–235.

Green, P. J.: 1995, 'Reversible jump Markov chain Monte Carlo computation and Bayesian model determination'. *Biometrika* **82**, 711–732.

Johnson, H., J. Martin, G. Foster, and R. Kuhn: 2007a, 'Improving Translation Quality by Discarding Most of the Phrasetable'. In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Prague, Czech Republic, pp. 967–975.

Koehn, P. and H. Hoang: 2007, 'Factored Translation Models'. In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Prague, Czech Republic, pp. 868–876.

Koehn, P., F. Och, and D. Marcu: 2003, 'Statistical phrase-based translation'. In: *Proc. of HLT-NAACL*. Morristown, NJ, USA, pp. 48–54.

Kumar, S. and W. Byrne: 2004, 'Minimum Bayes-Risk Decoding for Statistical Machine Translation'. In: D. M. Susan Dumais and S. Roukos (eds.): *HLT-NAACL 2004: Main Proceedings*. Boston, Massachusetts, USA, pp. 169–176, Association for Computational Linguistics.

Langlais, P., F. Gotti, and A. Patry: 2007, 'A Greedy Decoder for Phrase-Based Statistical Machine Translation'. In: *11th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 2007)*. SkŽvde, Sweden, pp. 104–113.

Li, Z., J. Eisner, and S. Khudanpur: 2009, 'Variational Decoding for Statistical Machine Translation'. In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. pp. 593–601.

Liu, D. C. and J. Nocedal: 1989, 'On the Limited Memory BFGS Method for Large Scale Optimization'. *Mathematical Programming* **45**(3), 503–528.

Marcu, D. and W. Wong: 2002, 'A phrase-based, joint probability model for statistical machine translation'. In: *EMNLP '02: Proceedings of the ACL-02 conference on Empirical methods in natural language processing*. Morristown, NJ, USA, pp. 133–139.

Metropolis, N. and S. Ulam: 1949, 'The Monte Carlo Method'. *Journal of the American Statistical Association* **44**(247), 335–341.

Och, F. J.: 2003, 'Minimum Error Rate Training in Statistical Machine Translation'. In: *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*. Sapporo, Japan, pp. 160–167.

Papineni, K., S. Roukos, T. Ward, and W.-J. Zhu: 2002, 'Bleu: a Method for Automatic Evaluation of Machine Translation'. In: *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA, pp. 311–318.

Schraudolph, N. N.: 1999, 'Local Gain Adaptation in Stochastic Gradient Descent'. Technical Report IDSIA-09-99, IDSIA.

Smith, D. A. and J. Eisner: 2006, 'Minimum Risk Annealing for Training Log-Linear Models'. In: *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*. Sydney, Australia, pp. 787–794.

Zens, R., S. Hasan, and H. Ney: 2007, 'A Systematic Comparison of Training Criteria for Statistical Machine Translation'. In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. pp. 524–532.