A Fast Fertility Hidden Markov Model for Word Alignment Using MCMC

Shaojun Zhao and **Daniel Gildea**
Department of Computer Science
University of Rochester

Abstract

A word in one language can be translated to zero, one, or several words in other languages. Using word fertility features has been shown to be useful in building word alignment models for statistical machine translation. We built a fertility hidden Markov model by adding fertility to the hidden Markov model. This model not only achieves lower alignment error rate than the hidden Markov model, but also runs faster. It is similar in some ways to IBM Model 4, but is much easier to understand. We use Gibbs sampling for parameter estimation, which is more principled than the neighborhood method used in IBM Model 4.

1 Introduction

IBM models and the hidden Markov model (HMM) for word alignment are the most influential statistical word alignment models (Brown et al., 1993; Vogel et al., 1996; Och and Ney, 2003). There are three kinds of important information for word alignment models: **lexicality**, **locality** and **fertility**. IBM Model 1 uses only lexical information; IBM Model 2 and the hidden Markov model take advantage of both lexical and locality information; IBM Models 4 and 5 use all three kinds of information, and they remain the state of the art despite the fact that they were developed almost two decades ago.

Recent experiments on large datasets have shown that the performance of the hidden Markov model is very close to IBM Model 4. Nevertheless, we believe that IBM Model 4 is essentially a better model because it exploits the fertility of words in the target language. However, IBM Model 4 is so complex that most researches use the GIZA++ software package (Och and Ney, 2003), and IBM Model 4 itself is treated as a black box. The complexity in IBM Model 4 makes it hard to understand and to improve. Our goal is to build a model that includes lexicality, locality, and fertility; and, at the same time, to make it easy to understand. We also want it to be accurate and computationally efficient.

There have been many years of research on word alignment. Our work is different from others in essential ways. Most other researchers take either the HMM alignments (Liang et al., 2006) or IBM Model 4 alignments (Cherry and Lin, 2003) as input and perform post-processing, whereas our model is a potential replacement for the HMM and IBM Model 4. Directly modeling fertility makes our model fundamentally different from others. Most models have limited ability to model fertility. Liang et al. (2006) learn the alignment in both translation directions jointly, essentially pushing the fertility towards 1. ITG models (Wu, 1997) assume the fertility to be either zero or one. It can model phrases, but the phrase has to be contiguous. There have been works that try to simulate fertility using the hidden Markov model (Toutanova et al., 2002; Deng and Byrne, 2005), but we prefer to model fertility directly.

Our model is a coherent generative model that combines the HMM and IBM Model 4. It is easier to understand than IBM Model 4 (see Section 3). Our model also removes several undesired properties in IBM Model 4. We use Gibbs sampling instead of a heuristic-based neighborhood method for parameter

estimation. Our distortion parameters are similar to IBM Model 2 and the HMM, while IBM Model 4 uses inverse distortion (Brown et al., 1993). Our model assumes that fertility follows a Poisson distribution, while IBM Model 4 assumes a multinomial distribution, and has to learn a much larger number of parameters, which makes it slower and less reliable. Our model is much faster than IBM Model 4. In fact, we will show that it is also faster than the HMM, and has lower alignment error rate than the HMM.

Parameter estimation for word alignment models that model fertility is more difficult than for models without fertility. Brown et al. (1993) and Och and Ney (2003) first compute the Viterbi alignments for simpler models, then consider only some neighbors of the Viterbi alignments for modeling fertility. If the optimal alignment is not in those neighbors, this method will not be able find the optimal alignment. We use the Markov Chain Monte Carlo (MCMC) method for training and decoding, which has nice probabilistic guarantees. DeNero et al. (2008) applied the Markov Chain Monte Carlo method to word alignment for machine translation; they do not model word fertility.

2 Statistical Word Alignment Models

2.1 Alignment and Fertility

Given a source sentence $\mathbf{f}_1^J = f_1, f_2, \ldots, f_J$ and a target sentence $\mathbf{e}_1^I = e_1, e_2, \ldots, e_I$, we define the alignments between the two sentences as a subset of the Cartesian product of the word positions. Following Brown et al. (1993), we assume that each source word is aligned to exactly one target word. We denote as $\mathbf{a}_1^J = a_1, a_2, \ldots, a_J$ the alignments between $\mathbf{f}_1^J$ and $\mathbf{e}_1^I$. When a word f_j is not aligned with any word e, a_j is 0. For convenience, we add an empty word ϵ to the target sentence at position 0 (i.e., $e_0 = \epsilon$). However, as we will see, we have to add more than one empty word for the HMM. In order to compute the "jump probability" in the HMM model, we need to know the position of the aligned target word for the previous source word. If the previous source word aligns to an empty word, we could use the position of the empty word to indicate the nearest previous source word that does not align to an empty word. For this reason, we use a total of $I + 1$ empty words for the HMM model[1]. Moore (2004) also suggested adding multiple empty words to the target sentence for IBM Model 1. After we add $I+1$ empty words to the target sentence, the alignment is a mapping from source to target word positions:

$$a : j \rightarrow i, i = a_j$$

where $j = 1, 2, \ldots, J$ and $i = 1, 2, \ldots, 2I + 1$. Words from position $I + 1$ to $2I + 1$ in the target sentence are all empty words.

We allow each source word to align with exactly one target word, but each target word may align with multiple source words.

The fertility ϕ_i of a word e_i at position i is defined as the number of aligned source words:

$$\phi_i = \sum_{j=1}^{J} \delta(a_j, i)$$

where δ is the Kronecker delta function:

$$\delta(x, y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{otherwise} \end{cases}$$

In particular, the fertility of all empty words in the target sentence is $\sum_{i=I+1}^{2I+1} \phi_i$. We define $\phi_\epsilon \equiv \sum_{i=I+1}^{2I+1} \phi_i$. For a bilingual sentence pair $\mathbf{e}_1^{2I+1}$ and $\mathbf{f}_1^J$, we have $\sum_{i=1}^{I} \phi_i + \phi_\epsilon = J$.

The inverted alignments for position i in the target sentence are a set B_i, such that each element in B_i is aligned with i, and all alignments of i are in B_i. Inverted alignments are explicitly used in IBM Models 3, 4 and 5, but not in our model, which is one reason that our model is easier to understand.

2.2 IBM Model 1 and HMM

IBM Model 1 and the HMM are both generative models, and both start by defining the probability of alignments and source sentence given the target sentence: $P(\mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1})$; the data likelihood can be computed by summing over alignments:

[1] If f_{j-1} does not align with an empty word and f_j aligns with an empty word, we want to record the position of the target word that f_{j-1} aligns with. There are $I + 1$ possibilities: f_j is the first word in the source sentence, or f_{j-1} aligns with one of the target word.

$P(\mathbf{f}_1^J|\mathbf{e}_1^{2I+1}) = \sum_{\mathbf{a}_1^J} P(\mathbf{a}_1^J, \mathbf{f}_1^J|\mathbf{e}_1^{2I+1})$. The alignments $\mathbf{a}_1^J$ are the hidden variables. The expectation maximization algorithm is used to learn the parameters such that the data likelihood is maximized.

Without loss of generality, $P(\mathbf{a}_1^J, \mathbf{f}_1^J|\mathbf{e}_1^{2I+1})$ can be decomposed into **length probabilities**, **distortion probabilities** (also called alignment probabilities), and **lexical probabilities** (also called translation probabilities):

$$\begin{aligned} & P(\mathbf{a}_1^J, \mathbf{f}_1^J|\mathbf{e}_1^{2I+1}) \\ = \;& P(J|\mathbf{e}_1^{2I+1}) \prod_{j=1}^{J} P(a_j, f_j|f_1^{j-1}, a_1^{j-1}, \mathbf{e}_1^{2I+1}) \\ = \;& P(J|\mathbf{e}_1^{2I+1}) \prod_{j=1}^{J} \Big(P(a_j|f_1^{j-1}, a_1^{j-1}, \mathbf{e}_1^{2I+1}) \times \\ & P(f_j|f_1^{j-1}, a_1^{j}, \mathbf{e}_1^{2I+1}) \Big) \end{aligned}$$

where $P(J|\mathbf{e}_1^{2I+1})$ is a length probability, $P(a_j|f_1^{j-1}, a_1^{j-1}, \mathbf{e}_1^{2I+1})$ is a distortion probability and $P(f_j|f_1^{j-1}, a_1^{j}, \mathbf{e}_1^{2I+1})$ is a lexical probability.

IBM Model 1 assumes a uniform distortion probability, a length probability that depends only on the length of the target sentence, and a lexical probability that depends only on the aligned target word:

$$P(\mathbf{a}_1^J, \mathbf{f}_1^J|\mathbf{e}_1^{2I+1}) = \frac{P(J|I)}{(2I+1)^J} \prod_{j=1}^{J} P(f_j|e_{a_j})$$

The hidden Markov model assumes a length probability that depends only on the length of the target sentence, a distortion probability that depends only on the previous alignment and the length of the target sentence, and a lexical probability that depends only on the aligned target word:

$$\begin{aligned} & P(\mathbf{a}_1^J, \mathbf{f}_1^J|\mathbf{e}_1^{2I+1}) = \\ & \qquad P(J|I) \prod_{j=1}^{J} P(a_j|a_{j-1}, I) P(f_j|e_{a_j}) \end{aligned}$$

In order to make the HMM work correctly, we enforce the following constraints (Och and Ney, 2003):

$$\begin{aligned} P(i+I+1|i', I) &= p_0 \delta(i, i') \\ P(i+I+1|i'+I+1, I) &= p_0 \delta(i, i') \\ P(i|i'+I+1, I) &= P(i|i', I) \end{aligned}$$

where the first two equations imply that the probability of jumping to an empty word is either 0 or p_0, and the third equation implies that the probability of jumping from a non-empty word is the same as the probability of jumping from the corespondent empty word.

The absolute position in the HMM is not important, because we re-parametrize the distortion probability in terms of the distance between adjacent alignment points (Vogel et al., 1996; Och and Ney, 2003):

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

where $c(\,)$ is the count of jumps of a given distance.

In IBM Model 1, the word order does not matter. The HMM is more likely to align a source word to a target word that is adjacent to the previous aligned target word, which is more suitable than IBM Model 1 because adjacent words tend to form phrases.

For these two models, in theory, the fertility for a target word can be as large as the length of the source sentence. In practice, the fertility for a target word in IBM Model 1 is not very big except for rare target words, which can become a garbage collector, and align to many source words (Brown et al., 1993; Och and Ney, 2003; Moore, 2004). The HMM is less likely to have this garbage collector problem because of the alignment probability constraint. However, fertility is an inherent cross-language property and these two models cannot assign consistent fertility to words. This is our motivation for adding fertility to these two models, and we expect that the resulting models will perform better than the baseline models. Because the HMM performs much better than IBM Model 1, we expect that the fertility hidden Markov model will perform much better than the fertility IBM Model 1. Throughout the paper, "our model" refers to the fertility hidden Markov model.

Due to space constraints, we are unable to provide details for IBM Models 3, 4 and 5; see Brown et al. (1993) and Och and Ney (2003). But we want to point out that the locality property modeled in the HMM is missing in IBM Model 3, and is modeled invertedly in IBM Model 4. IBM Model 5 removes deficiency (Brown et al., 1993; Och and Ney, 2003)

from IBM Model 4, but it is computationally very expensive due to the larger number of parameters than IBM Model 4, and IBM Model 5 often provides no improvement on alignment accuracy.

3 Fertility Hidden Markov Model

Our fertility IBM Model 1 and fertility HMM are both generative models and start by defining the probability of fertilities (for each non-empty target word and all empty words), alignments, and the source sentence given the target sentence: $P(\phi_1^I, \phi_\epsilon, \mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1})$; the data likelihood can be computed by summing over fertilities and alignments: $P(\mathbf{f}_1^J | \mathbf{e}_1^{2I+1}) = \sum_{\phi_1^I, \phi_\epsilon, \mathbf{a}_1^J} P(\phi_1^I, \phi_\epsilon, \mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1})$.

The fertility for a non-empty word e_i is a random variable ϕ_i, and we assume ϕ_i follows a Poisson distribution $\text{Poisson}(\phi_i; \lambda(e_i))$. The sum of the fertilities of all the empty words (ϕ_ϵ) grows with the length of the target sentence. Therefore, we assume that ϕ_ϵ follows a Poisson distribution with parameter $I\lambda(\epsilon)$.

Now $P(\phi_1^I, \phi_\epsilon, \mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1})$ can be decomposed in the following way:

$$
\begin{aligned}
&P(\phi_1^I, \phi_\epsilon, \mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1}) \\
&\quad = \quad P(\phi_1^I | \mathbf{e}_1^{2I+1}) P(\phi_\epsilon | \phi_1^I, \mathbf{e}_1^{2I+1}) \times \\
&\qquad \prod_{j=1}^{J} P(a_j, f_j | f_1^{j-1}, a_1^{j-1}, \mathbf{e}_1^{2I+1}, \phi_1^I, \phi_\epsilon) \\
&\quad = \quad \prod_{i=1}^{I} \frac{\lambda(e_i)^{\phi_i} e^{-\lambda(e_i)}}{\phi_i!} \times \\
&\qquad \frac{(I\lambda(\epsilon))^{\phi_\epsilon} \, e^{-I\lambda(\epsilon)}}{\phi_\epsilon!} \times \\
&\qquad \prod_{j=1}^{J} \Big(P(a_j | f_1^{j-1}, a_1^{j-1}, \mathbf{e}_1^{2I+1}, \phi_1^I, \phi_\epsilon) \times \\
&\qquad\quad P(f_j | f_1^{j-1}, a_1^j, \mathbf{e}_1^{2I+1}, \phi_1^I, \phi_\epsilon) \Big)
\end{aligned}
$$

Superficially, we only try to model the length probability more accurately. However, we also enforce the fertility for the same target word across the corpus to be consistent. The expected fertility for a non-empty word e_i is $\lambda(e_i)$, and the expected fertility for all empty words is $I\lambda(\epsilon)$. Any fertility value has a non-zero probability, but fertility values that are further away from the mean have low probability. IBM Models 3, 4, and 5 use a multinomial distribution for fertility, which has a much larger number of parameters to learn. Our model has only one parameter for each target word, which can be learned more reliably.

In the fertility IBM Model 1, we assume that the distortion probability is uniform, and the lexical probability depends only on the aligned target word:

$$
\begin{aligned}
&P(\phi_1^I, \phi_\epsilon, \mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1}) \\
&\quad = \quad \prod_{i=1}^{I} \frac{\lambda(e_i)^{\phi_i} e^{-\lambda(e_i)}}{\phi_i!} \times \\
&\qquad \frac{(I\lambda(\epsilon))^{\phi_\epsilon} \, e^{-(I\lambda(\epsilon))}}{\phi_\epsilon!} \times \\
&\qquad \frac{1}{(2I+1)^J} \prod_{j=1}^{J} P(f_j | e_{a_j}) \qquad (1)
\end{aligned}
$$

In the fertility HMM, we assume that the distortion probability depends only on the previous alignment and the length of the target sentence, and that the lexical probability depends only on the aligned target word:

$$
\begin{aligned}
&P(\phi_1^I, \phi_\epsilon, \mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1}) \\
&\quad = \quad \prod_{i=1}^{I} \frac{\lambda(e_i)^{\phi_i} e^{-\lambda(e_i)}}{\phi_i!} \times \\
&\qquad \frac{(I\lambda(\epsilon))^{\phi_\epsilon} \, e^{-(I\lambda(\epsilon))}}{\phi_\epsilon!} \times \\
&\qquad \prod_{j=1}^{J} P(a_j | a_{j-1}, I) P(f_j | e_{a_j}) \qquad (2)
\end{aligned}
$$

When we compute $P(\mathbf{f}_1^J | \mathbf{e}_1^{2I+1})$, we only sum over fertilities that agree with the alignments:

$$
P(\mathbf{f}_1^J | \mathbf{e}_1^{2I+1}) \quad = \quad \sum_{\mathbf{a}_1^J} P(\mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1})
$$

where

$$
\begin{aligned}
& P(\mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1}) \\
&= \sum_{\phi_1^I, \phi_\epsilon} P(\phi_1^I, \phi_\epsilon, \mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1}) \\
&\approx P(\phi_1^I, \phi_\epsilon, \mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1}) \times \\
&\quad \prod_{i=1}^{I} \delta \left(\sum_{j=1}^{J} \delta(a_j, i), \phi_i \right) \times \\
&\quad \delta \left(\sum_{i=I+1}^{2I+1} \sum_{j=1}^{J} \delta(a_j, i), \phi_\epsilon \right) \qquad (3)
\end{aligned}
$$

In the last two lines of Equation 3, ϕ_ϵ and each ϕ_i are not free variables, but are determined by the alignments. Because we only sum over fertilities that are consistent with the alignments, we have $\sum_{\mathbf{f}_1^J} P(\mathbf{f}_1^J | \mathbf{e}_1^{2I+1}) < 1$, and our model is deficient, similar to IBM Models 3 and 4 (Brown et al., 1993). We can remove the deficiency for fertility IBM Model 1 by assuming a different distortion probability: the distortion probability is 0 if fertility is not consistent with alignments, and uniform otherwise. The total number of consistent fertility and alignments is $\frac{J!}{\phi_\epsilon! \prod_{j=1}^{J} \phi_i!}$. Replacing $\frac{1}{(2I+1)^J}$ with $\frac{\phi_\epsilon! \prod_{j=1}^{J} \phi_i!}{J!}$, we have:

$$
\begin{aligned}
& P(\phi_1^I, \phi_\epsilon, \mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1}) \\
&= \prod_{i=1}^{I} \lambda(e_i)^{\phi_i} e^{-\lambda(e_i)} \times \\
&\quad (I\lambda(\epsilon))^{\phi_\epsilon} e^{-(I\lambda(\epsilon))} \times \\
&\quad \frac{1}{J!} \prod_{j=1}^{J} P(f_j | e_{a_j})
\end{aligned}
$$

In our experiments, we did not find a noticeable change in terms of alignment accuracy by removing the deficiency.

4 Expectation Maximization Algorithm

We estimate the parameters by maximizing $P(\mathbf{f}_1^J | \mathbf{e}_1^{2I+1})$ using the expectation maximization (EM) algorithm (Dempster et al., 1977). The auxiliary function is:

$$
\begin{aligned}
& L(P(f|e), P(a|a'), \lambda(e), \xi_1(e), \xi_2(a')) \\
&= \sum_{\mathbf{a}_1^J} \tilde{P}(\mathbf{a}_1^J | \mathbf{e}_1^{2I+1}, \mathbf{f}_1^J) \log P(\mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1}) \\
&\quad - \sum_e \xi_1(e) (\sum_f P(f|e) - 1) \\
&\quad - \sum_{a'} \xi_2(a') (\sum_a P(a|a') - 1)
\end{aligned}
$$

Because $P(\mathbf{a}_1^J, \mathbf{f}_1^J | \mathbf{e}_1^{2I+1})$ is in the exponential family, we get a closed form for the parameters from expected counts:

$$
P(f|e) = \frac{\sum_s c(f|e; \mathbf{f}^{(s)}, \mathbf{e}^{(s)})}{\sum_f \sum_s c(f|e; \mathbf{f}^{(s)}, \mathbf{e}^{(s)})} \qquad (4)
$$

$$
P(a|a') = \frac{\sum_s c(a|a'; \mathbf{f}^{(s)}, \mathbf{e}^{(s)})}{\sum_a \sum_s c(a|a'; \mathbf{f}^{(s)}, \mathbf{e}^{(s)})} \qquad (5)
$$

$$
\lambda(e) = \frac{\sum_s c(\phi|e; \mathbf{f}^{(s)}, \mathbf{e}^{(s)})}{\sum_s c(k|e; \mathbf{f}^{(s)}, \mathbf{e}^{(s)})} \qquad (6)
$$

where s is the number of bilingual sentences, and

$$
\begin{aligned}
c(f|e; \mathbf{f}_1^J, \mathbf{e}_1^{2I+1}) &= \sum_{\mathbf{a}_1^J} \tilde{P}(\mathbf{a}_1^J | \mathbf{f}_1^J, \mathbf{e}_1^{2I+1}) \times \\
&\quad \sum_j \delta(f_j, f) \delta(e_i, e) \\
c(a|a'; \mathbf{f}_1^J, \mathbf{e}_1^{2I+1}) &= \sum_{\mathbf{a}_1^J} \tilde{P}(\mathbf{a}_1^J | \mathbf{f}_1^J, \mathbf{e}_1^{2I+1}) \times \\
&\quad \sum_j \delta(a_j, a) \delta(a_{j-1}, a') \\
c(\phi|e; \mathbf{f}_1^J, \mathbf{e}_1^{2I+1}) &= \sum_{\mathbf{a}_1^J} \tilde{P}(\mathbf{a}_1^J | \mathbf{f}_1^J, \mathbf{e}_1^{2I+1}) \times \\
&\quad \sum_i \phi_i \delta(e_i, e) \\
c(k|e; \mathbf{f}_1^J, \mathbf{e}_1^{2I+1}) &= \sum_i k(e_i) \delta(e_i, e)
\end{aligned}
$$

These equations are for the fertility hidden Markov model. For the fertility IBM Model 1, we do not need to estimate the distortion probability.

5 Gibbs Sampling for Fertility HMM

Although we can estimate the parameters by using the EM algorithm, in order to compute the expected

counts, we have to sum over all possible alignments $\mathbf{a}_1^J$, which is, unfortunately, exponential. We developed a Gibbs sampling algorithm (Geman and Geman, 1984) to compute the expected counts.

For each target sentence $\mathbf{e}_1^{2I+1}$ and source sentence $\mathbf{f}_1^J$, we initialize the alignment a_j for each source word f_j using the Viterbi alignments from IBM Model 1. During the training stage, we try all $2I+1$ possible alignments for a_j but fix all other alignments.[2] We choose alignment a_j with probability $P(a_j|a_1, \cdots a_{j-1}, a_{j+1} \cdots a_J, \mathbf{f}_1^J, \mathbf{e}_1^{2I+1})$, which can be computed in the following way:

$$P(a_j|a_1, \cdots, a_{j-1}, a_{j+1}, \cdots, a_J, \mathbf{f}_1^J, \mathbf{e}_1^{2I+1}) = \frac{P(\mathbf{a}_1^J, \mathbf{f}_1^J|\mathbf{e}_1^{2I+1})}{\sum_{a_j} P(\mathbf{a}_1^J, \mathbf{f}_1^J|\mathbf{e}_1^{2I+1})} \quad (7)$$

For each alignment variable a_j, we choose t samples. We scan through the corpus many times until we are satisfied with the parameters we learned using Equations 4, 5, and 6. This Gibbs sampling method updates parameters constantly, so it is an "online learning" algorithm. However, this sampling method needs a large amount of communication between machines in order to keep the parameters up to date if we compute the expected counts in parallel. Instead, we do "batch learning": we fix the parameters, scan through the entire corpus and compute expected counts in parallel (E-step); then combine all the counts together and update the parameters (M-step). This is analogous to what IBM models and the HMM do in the EM algorithms. The algorithm for the E-step on one machine (all machines are independent) is in Algorithm 1.

For the fertility hidden Markov model, updating $P(\mathbf{a}_1^J, \mathbf{f}_1^J|\mathbf{e}_1^{2I+1})$ whenever we change the alignment a_j can be done in constant time, so the complexity of choosing t samples for all a_j $(j = 1, 2, \ldots, J)$ is $O(tIJ)$. This is the same complexity as the HMM if t is $O(I)$, and it has lower complexity if t is a constant. Surprisingly, we can achieve better results than the HMM by computing as few as 1 sample for each alignment, so the fertility hidden Markov model is much faster than the HMM. Even when choosing t such that our model is 5 times faster than the HMM, we achieve better results.

[2]For fertility IBM Model 1, we only need to compute $I+1$ values because e_{I+1}^{2I+1} are identical empty words.

Algorithm 1: One iteration of E-step: draw t samples for each a_j for each sentence pair $(\mathbf{f}_1^J, \mathbf{e}_1^{2I+1})$ in the corpus

```
for (f_1^J, e_1^{2I+1}) in the corpus do
    Initialize a_1^J with IBM Model 1;
    for t do
        for j do
            for i do
                a_j = i;
                Compute P(a_1^J, f_1^J | e_1^{2I+1});
            end
            Draw a sample for a_j using
            Equation 7;
            Update counts;
        end
    end
end
```

We also consider initializing the alignments using the HMM Viterbi algorithm in the E-step. In this case, the fertility hidden Markov model is not faster than the HMM. Fortunately, initializing using IBM Model 1 Viterbi does not decrease the accuracy in any noticeable way, and reduces the complexity of the Gibbs sampling algorithm.

In the testing stage, the sampling algorithm is the same as above except that we keep the alignments $\mathbf{a}_1^J$ that maximize $P(\mathbf{a}_1^J, \mathbf{f}_1^J|\mathbf{e}_1^{2I+1})$. We need more samples in the testing stage because it is unlikely to get to the optimal alignments by sampling a few times for each alignment. On the contrary, in the above training stage, although the samples are not accurate enough to represent the distribution defined by Equation 7 for each alignment a_j, it is accurate enough for computing the expected counts, which are defined at the corpus level. Interestingly, we found that throwing away the fertility and using the HMM Viterbi decoding achieves same results as the sampling approach (we can ignore the difference because it is tiny), but is faster. Therefore, we use Gibbs sampling for learning and the HMM Viterbi decoder for testing.

Gibbs sampling for the fertility IBM Model 1 is similar but simpler. We omit the details here.

Alignment	Model	P	R	AER
en → cn	IBM1	49.6	55.3	47.8
	IBM1F	55.4	57.1	43.8
	HMM	62.6	59.5	39.0
	HMMF-1	65.4	59.1	37.9
	HMMF-5	66.8	60.8	36.2
	HMMF-30	67.8	62.3	34.9
	IBM4	66.8	64.1	34.5
cn → en	IBM1	52.6	53.7	46.9
	IBM1F	55.9	56.4	43.9
	HMM	66.1	62.1	35.9
	HMMF-1	68.6	60.2	35.7
	HMMF-5	71.1	62.2	33.5
	HMMF-30	71.1	62.7	33.2
	IBM4	69.3	68.5	31.1

Table 1: AER results. IBM1F refers to the fertility IBM1 and HMMF refers to the fertility HMM. We choose $t = 1$, 5, and 30 for the fertility HMM.

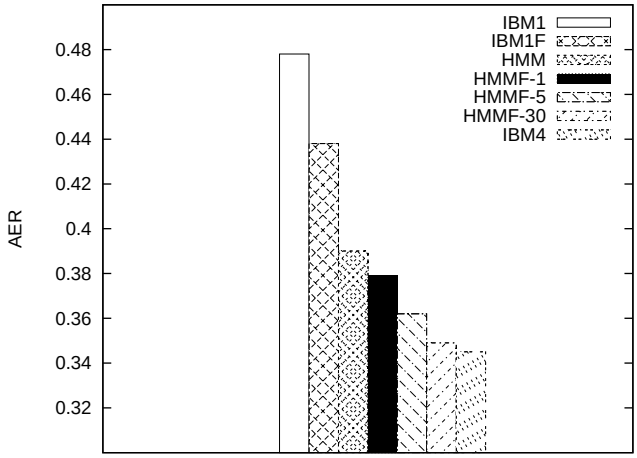

Figure 1: AER comparison (en→cn)

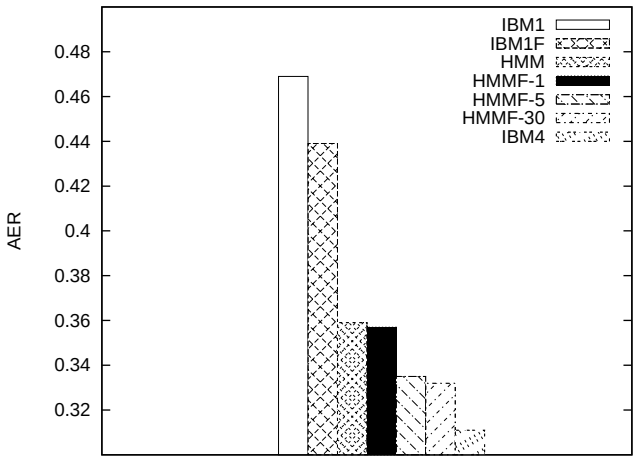

Figure 2: AER comparison (cn →en)

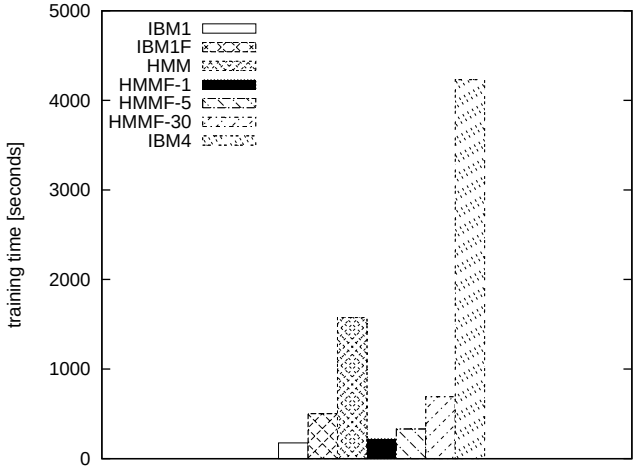

Figure 3: Training time comparison. The training time for each model is calculated from scratch. For example, the training time of IBM Model 4 includes the training time of IBM Model 1, the HMM, and IBM Model 3.

6 Experiments

We evaluated our model by computing the word alignment and machine translation quality. We use the alignment error rate (AER) as the word alignment evaluation criterion. Let A be the alignments output by word alignment system, P be a set of possible alignments, and S be a set of sure alignments both labeled by human beings. S is a subset of P.

Precision, recall, and AER are defined as follows:

$$\begin{aligned} \text{recall} &= \frac{|A \cap S|}{|S|} \\ \text{precision} &= \frac{|A \cap P|}{|A|} \\ \text{AER}(S, P, A) &= 1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|} \end{aligned}$$

AER is an extension to F-score. Lower AER is better.

We evaluate our fertility models on a Chinese-English corpus. The Chinese-English data taken from FBIS newswire data, and has 380K sentence pairs, and we use the first 100K sentence pairs as our training data. We used hand-aligned data as reference. The Chinese-English data has 491 sentence pairs.

We initialize IBM Model 1 and the fertility IBM Model 1 with a uniform distribution. We smooth all parameters ($\lambda(e)$ and $P(f|e)$) by adding a small value (10^{-8}), so they never become too small. We run both models for 5 iterations. AER results are computed using the IBM Model 1 Viterbi alignments, and the Viterbi alignments obtained from the Gibbs sampling algorithm.

We initialize the HMM and the fertility HMM with the parameters learned in the 5th iteration of IBM Model 1. We smooth all parameters ($\lambda(e)$, $P(a|a')$ and $P(f|e)$) by adding a small value (10^{-8}). We run both models for 5 iterations. AER results are computed using traditional HMM Viterbi decoding for both models.

It is always difficult to determine how many samples are enough for sampling algorithms. However, both fertility models achieve better results than their baseline models using a small amount of samples. For the fertility IBM Model 1, we sample 10 times for each a_j, and restart 3 times in the training stage; we sample 100 times and restart 12 times in the testing stage. For the fertility HMM, we sample 30 times for each a_j with no restarting in the training stage; no sampling in the testing stage because we use traditional HMM Viterbi decoding for testing. More samples give no further improvement.

Initially, the fertility IBM Model 1 and fertility HMM did not perform well. If a target word e only appeared a few times in the training corpus, our model cannot reliably estimate the parameter $\lambda(e)$. Hence, smoothing is needed. One may try to solve it by forcing all these words to share a same parameter $\lambda(e_{\text{infrequent}})$. Unfortunately, this does not solve the problem because all infrequent words tend to have larger fertility than they should. We solve the problem in the following way: estimate the parameter $\lambda(e_{\text{non\_empty}})$ for all non-empty words, all infrequent words share this parameter. We consider words that appear less than 10 times as infrequent words.

Table 1, Figure 1, and Figure 2 shows the AER results for different models. We can see that the fertility IBM Model 1 consistently outperforms IBM Model 1, and the fertility HMM consistently outperforms the HMM.

The fertility HMM not only has lower AER than the HMM, it also runs faster than the HMM. Figure 3 show the training time for different models. In fact, with just 1 sample for each alignment, our model archives lower AER than the HMM, and runs more than 5 times faster than the HMM. It is possible to use sampling instead of dynamic programming in the HMM to reduce the training time with no decrease in AER (often an increase). We conclude that the fertility HMM not only has better AER results, but also runs faster than the hidden Markov model.

We also evaluate our model by computing the machine translation BLEU score (Papineni et al., 2002) using the Moses system (Koehn et al., 2007). The training data is the same as the above word alignment evaluation bitexts, with alignments for each model symmetrized using the grow-diag-final heuristic. Our test is 633 sentences of up to length 50, with four references. Results are shown in Table 2; we see that better word alignment results do not lead to better translations.

Model	BLEU
HMM	19.55
HMMF-30	19.26
IBM4	18.77

Table 2: BLEU results

7 Conclusion

We developed a fertility hidden Markov model that runs faster and has lower AER than the HMM. Our model is thus much faster than IBM Model 4. Our model is also easier to understand than IBM Model 4. The Markov Chain Monte Carlo method used in our model is more principled than the heuristic-based neighborhood method in IBM Model 4. While better word alignment results do not necessarily correspond to better translation quality, our translation results are comparable in translation quality to both the HMM and IBM Model 4.

Acknowledgments We would like to thank Tagyoung Chung, Matt Post, and the anonymous reviewers for helpful comments. This work was supported by NSF grants IIS-0546554 and IIS-0910611.

References

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

Colin Cherry and Dekang Lin. 2003. A probability model to improve word alignment. In *Proceedings of ACL-03*.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the *EM* algorithm. *Journal of the Royal Statistical Society*, 39(1):1–21.

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

Yonggang Deng and William Byrne. 2005. HMM word and phrase alignment for statistical machine translation. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 169–176, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.

Stuart Geman and Donald Geman. 1984. Stochastic relaxation, Gibbs distribution, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, November.

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 *Proceedings of ACL, Demonstration Session*, pages 177–180.

P. Liang, B. Taskar, and D. Klein. 2006. Alignment by agreement. In *North American Association for Computational Linguistics (NAACL)*, pages 104–111.

Robert C. Moore. 2004. Improving IBM word alignment Model 1. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 518–525, Barcelona, Spain, July.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of ACL-02*.

Kristina Toutanova, H. Tolga Ilhan, and Christopher D. Manning. 2002. Extensions to HMM-based statistical word alignment models. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 87–94.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING-96*, pages 836–841.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.