Scalable Decipherment for Machine Translation via Hash Sampling

Sujith Ravi
Google
Mountain View, CA 94043
sravi@gooogle.com

Abstract

In this paper, we propose a new Bayesian inference method to train statistical machine translation systems using only non-parallel corpora. Following a probabilistic *decipherment* approach, we first introduce a new framework for decipherment training that is flexible enough to incorporate any number/type of features (besides simple bag-of-words) as side-information used for estimating translation models. In order to perform fast, efficient Bayesian inference in this framework, we then derive a *hash sampling* strategy that is inspired by the work of Ahmed et al. (2012). The new translation hash sampler enables us to scale elegantly to complex models (for the first time) and large vocabulary/corpora sizes. We show empirical results on the OPUS data—our method yields the best BLEU scores compared to existing approaches, while achieving significant computational speedups (several orders faster). We also report for the first time—BLEU score results for a large-scale MT task using only non-parallel data (EMEA corpus).

1 Introduction

Statistical machine translation (SMT) systems these days are built using large amounts of bilingual parallel corpora. The parallel corpora are used to estimate translation model parameters involving word-to-word translation tables, fertilities, distortion, phrase translations, syntactic transformations, etc. But obtaining parallel data is an expensive process and not available for all language pairs or domains. On the other hand, monolingual data (in written form) exists and is easier to obtain for many languages. Learning translation models from monolingual corpora could help address the challenges faced by modern-day MT systems, especially for low resource language pairs. Recently, this topic has been receiving increasing attention from researchers and new methods have been proposed to train statistical machine translation models using only monolingual data in the source and target language. The underlying motivation behind most of these methods is that statistical properties for linguistic elements are shared across different languages and some of these similarities (mappings) could be automatically identified from large amounts of monolingual data.

The MT literature does cover some prior work on extracting or augmenting partial lexicons using non-parallel corpora (Rapp, 1995; Fung and McKeown, 1997; Koehn and Knight, 2000; Haghighi et al., 2008). However, none of these methods attempt to train end-to-end MT models, instead they focus on mining bilingual lexicons from monolingual corpora and often they require parallel seed lexicons as a starting point. Some of them (Haghighi et al., 2008) also rely on additional linguistic knowledge such as orthography, etc. to mine word translation pairs across related languages (e.g., Spanish/English). Unsupervised training methods have also been proposed in the past for related problems in decipherment (Knight and Yamada, 1999; Snyder et al., 2010; Ravi and Knight, 2011a) where the goal is to decode unknown scripts or ciphers.

The body of work that is more closely related to ours include that of Ravi and Knight (2011b) who introduced a *decipherment* approach for training translation models using only monolingual cor-

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 362–371,
Sofia, Bulgaria, August 4-9 2013.

pora. Their best performing method uses an EM algorithm to train a word translation model and they show results on a Spanish/English task. Nuhn et al. (2012) extend the former approach and improve training efficiency by pruning translation candidates prior to EM training with the help of context similarities computed from monolingual corpora.

In this work we propose a new Bayesian inference method for estimating translation models from scratch using only monolingual corpora. Secondly, we introduce a new feature-based representation for sampling translation candidates that allows one to incorporate any amount of additional features (beyond simple bag-of-words) as side-information during decipherment training. Finally, we also derive a new accelerated sampling mechanism using locality sensitive hashing inspired by recent work on fast, probabilistic inference for unsupervised clustering (Ahmed et al., 2012). The new sampler allows us to perform fast, efficient inference with more complex translation models (than previously used) and scale better to large vocabulary and corpora sizes compared to existing methods as evidenced by our experimental results on two different corpora.

2 Decipherment Model for Machine Translation

We now describe the decipherment problem formulation for machine translation.

Problem Formulation: Given a source text f (i.e., source word sequences $f_1 ... f_m$) and a monolingual target language corpus, our goal is to decipher the source text and produce a target translation.

Contrary to standard machine translation training scenarios, here we have to estimate the translation model $P_\theta(f|e)$ parameters using only monolingual data. During decipherment training, our objective is to estimate the model parameters in order to maximize the probability of the source text f as suggested by Ravi and Knight (2011b).

$$\arg\max_{\theta} \prod_{f} \sum_{e} P(e) \cdot P_\theta(f|e) \qquad (1)$$

For $P(e)$, we use a word n-gram language model (LM) trained on monolingual target text. We then estimate the parameters of the translation model $P_\theta(f|e)$ during training.

Translation Model: Machine translation is a much more complex task than solving other decipherment tasks such as word substitution ciphers (Ravi and Knight, 2011b; Dou and Knight, 2012). The mappings between languages involve non-determinism (i.e., words can have multiple translations), re-ordering of words can occur as grammar and syntax varies with language, and in addition word insertion and deletion operations are also involved.

Ideally, for the translation model $P(f|e)$ we would like to use well-known statistical models such as IBM Model 3 and estimate its parameters θ using the EM algorithm (Dempster et al., 1977). But training becomes intractable with complex translation models and scalability is also an issue when large corpora sizes are involved and the translation tables become huge to fit in memory. So, instead we use a simplified generative process for the translation model as proposed by Ravi and Knight (2011b) and used by others (Nuhn et al., 2012) for this task:

1. Generate a target (e.g., English) string $e = e_1 ... e_l$, with probability $P(e)$ according to an n-gram language model.

2. Insert a NULL word at any position in the English string, with uniform probability.

3. For each target word token e_i (including NULLs), choose a source word translation f_i, with probability $P_\theta(f_i|e_i)$. The source word may be NULL.

4. Swap any pair of adjacent source words f_{i-1}, f_i, with probability $P(swap)$; set to 0.1.

5. Output the foreign string $f = f_1 ... f_m$, skipping over NULLs.

Previous approaches (Ravi and Knight, 2011b; Nuhn et al., 2012) use the EM algorithm to estimate all the parameters θ in order to maximize likelihood of the foreign corpus. Instead, we propose a new Bayesian inference framework to estimate the translation model parameters. In spite of using Bayesian inference which is typically slow in practice (with standard Gibbs sampling), we show later that our method is scalable and permits decipherment training using more complex translation models (with several additional parameters).

2.1 Adding Phrases, Flexible Reordering and Fertility to Translation Model

We now extend the generative process (described earlier) to more complex translation models.

Non-local Re-ordering: The generative process described earlier limits re-ordering to local or adjacent word pairs in a source sentence. We extend this to allow re-ordering between any pair of words in the sentence.

Fertility: We also add a fertility model $P_{\theta_{fert}}$ to the translation model using the formula:

$$P_{\theta_{fert}} = \prod_i n_\theta(\phi_i|e_i) \cdot p_1^{\phi_0} \quad (2)$$

$$n_\theta(\phi_i|e_i) = \frac{\alpha_{fert} \cdot P_0(\phi_i|e_i) + C^{-i}(e_i, \phi_i)}{\alpha_{fert} + C^{-i}(e_i)} \quad (3)$$

where, P_0 represents the base distribution (which is set to uniform) in a Chinese Restaurant Process (CRP)[1] for the fertility model and C^{-i} represents the count of events occurring in the history excluding the observation at position i. ϕ_i is the number of source words aligned to (i.e., generated by) the target word e_i. We use sparse Dirichlet priors for all the translation model components.[2] ϕ_0 represents the target NULL word fertility and p_1 is the insertion probability which is fixed to 0.1. In addition, we set a maximum threshold for fertility values $\phi_i \leq \gamma \cdot m$, where m is the length of the source sentence. This discourages a particular target word (e.g., NULL word) from generating too many source words in the same sentence. In our experiments, we set $\gamma = 0.3$. We enforce this constraint in the training process during sampling.[3]

Modeling Phrases: Finally, we extend the translation candidate set in $P_\theta(f_i|e_i)$ to model phrases in addition to words for the target side (i.e., e_i can now be a word or a phrase[4] previously seen in the monolingual target corpus). This greatly increases the training time since in each sampling step, we now have many more e_i candidates to choose from. In Section 4, we describe how we deal with this problem by using a fast, efficient sampler based on hashing that allows us to speed up the Bayesian inference significantly whereas standard Gibbs sampling would be extremely slow.

[1] Each component in the translation model (word/phrase translations $P_\theta(f_i|e_i)$, fertility $P_{\theta_{fert}}$, etc.) is modeled using a CRP formulation.

[2] i.e., All the concentration parameters are set to low values; $\alpha_{f|e} = \alpha_{fert}$ = 0.01.

[3] We only apply this constraint when training on source text/corpora made of long sentences (>10 words) where the sampler might converge very slowly. For short sentences, a sparse prior on fertility α_{fert} typically discourages a target word from being aligned to too many different source words.

[4] Phrase size is limited to two words in our experiments.

3 Feature-based representation for Source and Target

The model described in the previous section while being flexible in describing the translation process, poses several challenges for training. As the source and target vocabulary sizes increase the size of the translation table ($|V_f| \cdot |V_e|$) increases significantly and often becomes too huge to fit in memory. Additionally, performing Bayesian inference with such a complex model using standard Gibbs sampling can be very slow in practice. Here, we describe a new method for doing Bayesian inference by first introducing a feature-based representation for the source and target words (or phrases) from which we then derive a novel proposal distribution for sampling translation candidates.

We represent both source and target words in a *vector space* similar to how documents are represented in typical information retrieval settings. But unlike documents, here each word $\mathbf{w}$ is associated with a feature vector $w^1 ... w^d$ (where w^i represents the weight for the feature indexed by i) which is constructed from monolingual corpora. For instance, *context features* for word $\mathbf{w}$ may include other words (or phrases) that appear in the immediate context (n-gram window) surrounding $\mathbf{w}$ in the monolingual corpus. Similarly, we can add other features based on *topic models*, *orthography* (Haghighi et al., 2008), *temporal* (Klementiev et al., 2012), etc. to our representation all of which can be extracted from monolingual corpora.

Next, given two high dimensional vectors $\mathbf{u}$ and $\mathbf{v}$ it is possible to calculate the similarity between the two words denoted by $s(\mathbf{u}, \mathbf{v})$. The feature construction process is described in more detail below:

Target Language: We represent each word (or phrase) e_i with the following contextual features along with their counts: (a) $f_{-context}$: every (word n-gram, position) pair immediately preceding e_i in the monolingual corpus (n=1, position=-1), (b) similar features $f_{+context}$ to model the context following e_i, and (c) we also throw in generic context features $f_{scontext}$ without position information—every word that co-occurs with e_i in the same sen-

tence. While the two position-features provide specific context information (may be sparse for large monolingual corpora), this feature is more generic and captures long-distance co-occurrence statistics.

Source Language: Words appearing in a source sentence f are represented using the corresponding target translation $e = e_1...e_m$ generated for f in the current sample during training. For each source word $f_j \in f$, we look at the corresponding word e_j in the target translation. We then extract all the context features of e_j in the target translation sample sentence e and add these features ($f_{-context}, f_{+context}, f_{scontext}$) with weights to the feature representation for f_j.

Unlike the target word feature vectors (which can be pre-computed from the monolingual target corpus), the feature vector for every source word f_j is dynamically constructed from the target translation sampled in each training iteration. This is a key distinction of our framework compared to previous approaches that use contextual similarity (or any other) features constructed from static monolingual corpora (Rapp, 1995; Koehn and Knight, 2000; Nuhn et al., 2012).

Note that as we add more and more features for a particular word (by training on larger monolingual corpora or adding new types of features, etc.), it results in the feature representation becoming more sparse (especially for source feature vectors) which can cause problems in efficiency as well as robustness when computing similarity against other vectors. In the next section, we will describe how we mitigate this problem by projecting into a low-dimensional space by computing hash signatures.

In all our experiments, we only use the features described above for representing source and target words. We note that the new sampling framework is easily extensible to many additional feature types (for example, monolingual topic model features, etc.) which can be efficiently handled by our inference algorithm and could further improve translation performance but we leave this for future work.

4 Bayesian MT Decipherment via Hash Sampling

The next step is to use the feature representations described earlier and iteratively sample a target word (or phrase) translation candidate e_i for every word f_i in the source text f. This involves choosing from $|V_e|$ possible target candidates in every step which can be highly inefficient (and infeasible for large vocabulary sizes). One possible strategy is to compute similarity scores $s(\mathbf{w_{f_i}}, \mathbf{w_{e'}})$ between the current source word feature vector $\mathbf{w_{f_i}}$ and feature vectors $\mathbf{w_{e' \in V_e}}$ for all possible candidates in the target vocabulary. Following this, we can prune the translation candidate set by keeping only the top candidates e^* according to the similarity scores. Nuhn et al. (2012) use a similar strategy to obtain a more compact translation table that improves runtime efficiency for EM training. Their approach requires calculating and sorting all $|V_e| \cdot |V_f|$ distances in time $O(V^2 \cdot log(V))$, where $V = max(|V_e|, |V_f|)$.

Challenges: Unfortunately, there are several additional challenges which makes inference very hard in our case. Firstly, we would like to include as many features as possible to represent the source/target words in our framework besides simple bag-of-words context similarity (for example, left-context, right-context, and other general-purpose features based on topic models, etc.). This makes the complexity far worse (in practice) since the dimensionality of the feature vectors d is a much higher value than $|V_e|$. Computing similarity scores alone (naïvely) would incur $O(|V_e| \cdot d)$ time which is prohibitively huge since we have to do this for every token in the source language corpus. Secondly, for Bayesian inference we need to sample from a distribution that involves computing probabilities for all the components (language model, translation model, fertility, etc.) described in Equation 1. This distribution needs to be computed for every source word token f_i in the corpus, for all possible candidates $e_i \in V_e$ and the process has to be repeated for multiple sampling iterations (typically more than 1000). Doing standard collapsed Gibbs sampling in this scenario would be very slow and intractable.

We now present an alternative fast, efficient inference strategy that overcomes many of the challenges described above and helps accelerate the sampling process significantly. First, we set our translation models within the context of a more generic and widely known family of distributions—mixtures of exponential families. Then we derive a novel proposal distribution for sampling translation candidates and introduce a new sampler for decipherment training that

is based on locality sensitive hashing (LSH).

Hashing methods such as LSH have been widely used in the past in several scenarios including NLP applications (Ravichandran et al., 2005). Most of these approaches employ LSH within heuristic methods for speeding up nearest-neighbor look up and similarity computation techniques. However, we use LSH hashing within a probabilistic framework which is very different from the typical use of LSH.

Our work is inspired by some recent work by Ahmed et al. (2012) on speeding up Bayesian inference for unsupervised clustering. We use a similar technique as theirs but a different approximate distribution for the proposal, one that is better-suited for machine translation models and without some of the additional overhead required for computing certain terms in the original formulation.

Mixtures of Exponential Families: The translation models described earlier (Section 2) can be represented as mixtures of exponential families, specifically mixtures of multinomials. In exponential families, distributions over random variables are given by:

$$p(x;\theta) = \exp(\langle \phi(x), \theta \rangle) - g(\theta) \tag{4}$$

where, $\phi : \mathcal{X} \rightarrow \mathcal{F}$ is a map from x to the space of sufficient statistics and $\theta \in \mathcal{F}$. The term $g(\theta)$ ensures that $p(x;\theta)$ is properly normalized. $\mathcal{X}$ is the domain of observations $X = x_1, ..., x_m$ drawn from some distribution p. Our goal is to estimate p. In our case, this refers to the translation model from Equation 1.

We also choose corresponding conjugate Dirichlet distributions for priors which have the property that the posterior distribution $p(\theta|X)$ over θ remains in the same family as $p(\theta)$.

Note that the (translation) model in our case consists of multiple exponential families components—a multinomial pertaining to the language model (which remains fixed[5]), and other components pertaining to translation probabilities $P_\theta(f_i|e_i)$, fertility $P_{\theta_{fert}}$, etc. To do collapsed Gibbs sampling under this model, we would perform the following steps during sampling:

1. For a given source word token f_i draw target translation

$$\begin{aligned} e_i \;\; &\sim \;\; p(e_i|F, E^{-i}) \\ &\propto p(e) \cdot p(f_i|e_i, F^{-i}, E^{-i}) \\ &\quad \cdot \; p_{fert}(\cdot|e_i, F^{-i}, E^{-i}) \cdot ... \end{aligned} \tag{5}$$

where, F is the full source text and E the full target translation generated during sampling.

2. Update the sufficient statistics for the changed target translation assignments.

For large target vocabularies, computing $p(f_i|e_i, F^{-i}, E^{-i})$ dominates the inference procedure. We can accelerate this step significantly using a good proposal distribution via hashing.

Locality Sensitive Hash Sampling: For general exponential families, here is a Taylor approximation for the data likelihood term (Ahmed et al., 2012):

$$p(x|\cdot) \approx \exp(\langle \phi(x), \theta^* \rangle) - g(\theta^*) \tag{6}$$

where, θ^* is the expected parameter (sufficient statistics).

For sampling the translation model, this involves computing an expensive inner product $\langle \phi(f_i), \theta^*_{e'} \rangle$ for each source word f_i which has to be repeated for every translation candidate e', including candidates that have very low probabilities and are unlikely to be chosen as the translation for f_j.

So, during decipherment training a standard collapsed Gibbs sampler will waste most of its time on expensive computations that will be discarded in the end anyways. Also, unlike some standard generative models used in other unsupervised learning scenarios (e.g., clustering) that model only observed features (namely words appearing in the document), here we would like to enrich the translation model with a lot more features (side-information).

Instead, we can accelerate the computation of the inner product $\langle \phi(f_i), \theta^*_{e'} \rangle$ using a *hash sampling* strategy similar to (Ahmed et al., 2012). The underlying idea here is to use binary hashing (Charikar, 2002) to explore only those candidates e' that are sufficiently close to the best matching translation via a proposal distribution. Next, we briefly introduce some notations and existing theoretical results related to binary hashing before describing the hash sampling procedure.

For any two vectors $u, v \in \mathbb{R}^n$,

$$\langle u, v \rangle = \|u\| \cdot \|v\| \cdot \cos \measuredangle(u, v) \tag{7}$$

[5] A high value for the LM concentration parameter α ensures that the LM probabilities do not deviate too far from the original fixed base distribution during sampling.

$$\measuredangle(u, v) = \pi Pr\{sgn[\langle u, w\rangle] \neq sgn[\langle v, w\rangle]\} \quad (8)$$

where, w is a random vector drawn from a symmetric spherical distribution and the term inside $Pr\{\cdot\}$ represents the relation between the signs of the two inner products.

Let $h^l(v) \in \{0,1\}^l$ be an l-bit binary hash of v where: $[h^l(v)]_i := sgn[\langle v, w_i\rangle]$; $w_i \sim U_m$. Then the probability of matching signs is given by:

$$z^l(u, v) := \frac{1}{l}\|h(u) - h(v)\|_1 \quad (9)$$

So, $z^l(u, v)$ measures how many bits differ between the hash vectors $h(u)$ and $h(v)$ associated with u, v. Combining this with Equations 6 and 7 we can estimate the unnormalized log-likelihood of a source word f_i being translated as target e' via:

$$s^l(f_i, e') \propto \|\theta_{e'}\| \cdot \|\phi(f_i)\| \cdot \cos \pi z^l(\phi(f_i), \theta_{e'}) \quad (10)$$

For each source word f_i, we now sample from this new distribution (after normalization) instead of the original one. The binary hash representation for the two vectors yield significant speedups during sampling since Hamming distance computation between $h(u)$ and $h(v)$ is highly optimized on modern CPUs. Hence, we can compute an estimate for the inner product quite efficiently.[6]

Updating the hash signatures: During training, we compute the target candidate projection $h(\theta_{e'})$ and corresponding norm only once[7] which is different from the setup of Ahmed et al. (2012). The source word projection $\phi(f_i)$ is dynamically updated in every sampling step. Note that doing this naïvely would scale slowly as $O(Dl)$ where D is the total number of features but instead we can update the hash signatures in a more efficient manner that scales as $O(D_{i_{>0}}l)$ where $D_{i_{>0}}$ is the number of non-zero entries in the feature representation for the source word $\phi(f_i)$. Also, we do not need to store the random vectors w in practice since these can be computed on the fly using hash functions. The inner product approximation also yields some theoretical guarantees for the hash sampler.[8]

[6] We set l = 32 bits in our experiments.

[7] In practice, we can ignore the *norm* terms to further speed up sampling since this is only an estimate for the proposal distribution and we follow this with the Metropolis Hastings step.

[8] For further details, please refer to (Ahmed et al., 2012).

4.1 Metropolis Hastings

In each sampling step, we use the distribution from Equation 10 as a proposal distribution in a Metropolis Hastings scheme to sample target translations for each source word.

Once a new target translation e' is sampled for source word f_i from the proposal distribution $q(\cdot) \propto \exp^{s^l(f_i, e')}$, we accept the proposal (and update the corresponding hash signatures) according to the probability r

$$r = \frac{q(e_i^{old}) \cdot p_{new}(\cdot)}{q(e_i^{new}) \cdot p_{old}(\cdot)} \quad (11)$$

where, $p_{old}(\cdot), p_{new}(\cdot)$ are the true conditional likelihood probabilities according to our model (including the language model component) for the old, new sample respectively.

5 Training Algorithm

Putting together all the pieces described in the previous section, we perform the following steps:

1. *Initialization:* We initialize the starting sample as follows: for each source word token, randomly sample a target word. If the source word also exists in the target vocabulary, then choose identity translation instead of the random one.[9]

2. *Hash Sampling Steps:* For each source word token f_i, run the hash sampler:

(a) Generate a proposal distribution by computing the hamming distance between the feature vectors for the source word and each target translation candidate. Sample a new target translation e_i for f_i from this distribution.

(b) Compute the acceptance probability for the chosen translation using a Metropolis Hastings scheme and accept (or reject) the sample. In practice, computation of the acceptance probability only needs to be done every r iterations (where r can be anywhere from 5 or 100).

Iterate through steps (2a) and (2b) for every word in the source text and then repeat this process for multiple iterations (usually 1000).

3. *Other Sampling Operators:* After every k iterations,[10] perform the following sampling operations:

(a) Re-ordering: For each source word token f_i at position i, randomly choose another position j

[9] Initializing with identity translation rather than random choice helps in some cases, especially for *unknown* words that involve named entities, etc.

[10] We set k = 3 in our experiments.

Corpus	Language	Sent.	Words	Vocab.
OPUS	Spanish	13,181	39,185	562
	English	19,770	61,835	411
EMEA	French	550,000	8,566,321	41,733
	Spanish	550,000	7,245,672	67,446

Table 1: Statistics of non-parallel corpora used here.

in the source sentence and swap the translations e_i with e_j. During the sampling process, we compute the probabilities for the two samples—the original and the swapped versions, and then sample an alignment from this distribution.

(b) Deletion: For each source word token, delete the current target translation (i.e., align it with the target NULL token). As with the reordering operation, we sample from a distribution consisting of the original and the deleted versions.

4. *Decoding the foreign sentence:* Finally, once the training is done (i.e., after all sampling iterations) we choose the final sample as our target translation output for the source text.

6 Experiments and Results

We test our method on two different corpora. To evaluate translation quality, we use BLEU score (Papineni et al., 2002), a standard evaluation measure used in machine translation.

First, we present MT results on non-parallel Spanish/English data from the OPUS corpus (Tiedemann, 2009) which was used by Ravi and Knight (2011b) and Nuhn et al. (2012). We show that our method achieves the best performance (BLEU scores) on this task while being significantly faster than both the previous approaches. We then apply our method to a much larger non-parallel French/Spanish corpus constructed from the EMEA corpus (Tiedemann, 2009). Here the vocabulary sizes are much larger and we show how our new Bayesian decipherment method scales well to this task inspite of using complex translation models. We also report the first BLEU results on such a large-scale MT task under truly non-parallel settings (without using any parallel data or seed lexicon).

For both the MT tasks, we also report BLEU scores for a baseline system using *identity* translations for common words (words appearing in both source/target vocabularies) and random translations for other words.

6.1 MT Task and Data

OPUS movie subtitle corpus (Tiedemann, 2009): This is a large open source collection of parallel corpora available for multiple language pairs. We use the same non-parallel Spanish/English corpus used in previous works (Ravi and Knight, 2011b; Nuhn et al., 2012). The details of the corpus are listed in Table 1. We use the entire Spanish source text for decipherment training and evaluate the final English output to report BLEU scores.

EMEA corpus (Tiedemann, 2009): This is a parallel corpus made out of PDF documents (articles from the medical domain) from the European Medicines Agency. We reserve the first 1k sentences in French as our source text (also used in decipherment training). To construct a non-parallel corpus, we split the remaining 1.1M lines as follows: first 550k sentences in French, last 550k sentences in Spanish. The latter is used to construct a target language model used for decipherment training. The corpus statistics are shown in Table 1.

6.2 Results

OPUS: We compare the MT results (BLEU scores) from different systems on the OPUS corpus in Table 2. The first row displays baseline performance. The next three rows 1a–1c display performance achieved by two methods from Ravi and Knight (2011b). Rows 2a, 2b show results from the of Nuhn et al. (2012). The last two rows display results for the new method using Bayesian hash sampling. Overall, using a 3-gram language model (instead of 2-gram) for decipherment training improves the performance for all methods. We observe that our method produces much better results than the others even with a 2-gram LM. With a 3-gram LM, the new method achieves the best performance; the highest BLEU score reported on this task. It is also interesting to note that the hash sampling method yields much better results than the Bayesian inference method presented in (Ravi and Knight, 2011b). This is due to the accelerated sampling scheme introduced earlier which helps it converge to better solutions faster.

Table 2 (last column) also compares the efficiency of different methods in terms of CPU time required for training. Both our 2-gram and 3-gram based methods are significantly faster than those previously reported for EM based training methods presented in (Ravi and Knight, 2011b; Nuhn

Method	BLEU	Time (hours)
Baseline system *(identity translations)*	6.9	
1a. EM with 2-gram LM (Ravi and Knight, 2011b)	15.3	∼850h
1b. EM with whole-segment LM (Ravi and Knight, 2011b)	19.3	
1c. Bayesian IBM Model 3 with 2-gram LM (Ravi and Knight, 2011b)	15.1	
2a. EM+Context with 2-gram LM (Nuhn et al., 2012)	15.2	50h
2b. EM+Context with 3-gram LM (Nuhn et al., 2012)	20.9	200h
3. Bayesian (standard) Gibbs sampling with 2-gram LM	-	222h
4a. Bayesian Hash Sampling* with 2-gram LM *(this work)*	**20.3**	2.6h
4b. Bayesian Hash Sampling* with 3-gram LM *(this work)*	**21.2**	2.7h
*(*sampler was run for 1000 iterations)*		

Table 2: Comparison of MT performance (BLEU scores) and efficiency (running time in CPU hours) on the Spanish/English OPUS corpus using only non-parallel corpora for training. For the Bayesian methods 4a and 4b, the samplers were run for 1000 iterations each on a single machine (1.8GHz Intel processor). For 1a, 2a, 2b, we list the training times as reported by Nuhn et al. (2012) based on their EM implementation for different settings.

Method	BLEU
Baseline system *(identity translations)*	3.0
Bayesian Hash Sampling with 2-gram LM	
vocab=*full* (V_e), add_fertility=*no*	4.2
vocab=*pruned**, add_fertility=*yes*	**5.3**

Table 3: MT results on the French/Spanish EMEA corpus using the new hash sampling method. *The last row displays results when we sample target translations from a *pruned* candidate set (most frequent 1k Spanish words + identity translation candidates) which enables the sampler to run much faster when using more complex models.

Spanish (***e***)		French (***f***)
el	→	les
la	→	la
por	→	des
sección	→	rubrique
administración	→	administration

Table 4: Sample (1-best) Spanish/French translations produced by the new method on the EMEA corpus using word translation models trained with non-parallel corpora.

et al., 2012). This is very encouraging since Nuhn et al. (2012) reported obtaining a speedup by pruning translation candidates (to ∼1/8th the original size) prior to EM training. On the other hand, we sample from the full set of translation candidates including additional target phrase (of size 2) candidates which results in a much larger vocabulary consisting of 1600 candidates (∼4 times the original size), yet our method runs much faster and yields better results. The table also demonstrates the siginificant speedup achieved by the hash sampler over a standard Gibbs sampler for the same model (∼85 times faster when using a 2-gram LM).

We also compare the results against MT performance from *parallel training*—MOSES system (Koehn et al., 2007) trained on 20k sentence pairs. The comparable number for Table 2 is 63.6 BLEU.

EMEA Results Table 3 shows the results achieved by our method on the larger task involving EMEA corpus. Here, the target vocabulary V_e is much higher (67k). In spite of this challenge and the model complexity, we can still perform decipherment training using Bayesian inference. We report the first BLEU score results on such a large-scale task using a 2-gram LM. This is achieved without using any seed lexicon or parallel corpora. The results are encouraging and demonstrates the ability of the method to scale to large-scale settings while performing efficient inference with complex models, which we believe will be especially useful for future MT application in scenarios where parallel data is hard to obtain. Table 4 displays some sample 1-best translations learned using this method.

For comparison purposes, we also evaluate MT performance on this task using *parallel training* (MOSES trained with hundred sentence pairs) and observe a BLEU score of 11.7.

7 Discussion and Future Work

There exists some work (Dou and Knight, 2012; Klementiev et al., 2012) that uses monolingual corpora to induce phrase tables, etc. These when combined with standard MT systems such as Moses (Koehn et al., 2007) trained on parallel corpora, have been shown to yield some BLEU score improvements. Nuhn et al. (2012) show some sample English/French lexicon entries learnt using EM algorithm with a pruned translation candidate set on a portion of the Gigaword corpus[11] but do not report any actual MT results. In addition, as we showed earlier our method can use Bayesian inference (which has a lot of nice properties compared to EM for unsupervised natural language tasks (Johnson, 2007; Goldwater and Griffiths, 2007)) and still scale easily to large vocabulary, data sizes while allowing the models to grow in complexity. Most importantly, our method produces better translation results (as demonstrated on the OPUS MT task). And to our knowledge, this is the first time that anyone has reported MT results under truly non-parallel settings on such a large-scale task (EMEA).

Our method is also easily extensible to out-of-domain translation scenarios similar to (Dou and Knight, 2012). While their work also uses Bayesian inference with a slice sampling scheme, our new approach uses a novel hash sampling scheme for decipherment that can easily scale to more complex models. The new decipherment framework also allows one to easily incorporate additional information (besides standard word translations) as features (e.g., context features, topic features, etc.) for unsupervised machine translation which can help further improve the performance in addition to accelerating the sampling process. We already demonstrated the utility of this system by going beyond words and incorporating phrase translations in a decipherment model for the first time.

In the future, we can obtain further speedups (especially for large-scale tasks) by parallelizing the sampling scheme seamlessly across multiple machines and CPU cores. The new framework can also be stacked with complementary techniques such as slice sampling, blocked (and type) sampling to further improve inference efficiency.

[11] http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2003T05

8 Conclusion

To summarize, our method is significantly faster than previous methods based on EM or Bayesian with standard Gibbs sampling and obtains better results than any previously published methods for the same task. The new framework also allows performing Bayesian inference for decipherment applications with more complex models than previously shown. We believe this framework will be useful for further extending MT models in the future to improve translation performance and for many other unsupervised decipherment application scenarios.

References

Amr Ahmed, Sujith Ravi, Shravan Narayanamurthy, and Alex Smola. 2012. Fastex: Hash clustering with exponential families. In *Proceedings of the 26th Conference on Neural Information Processing Systems (NIPS)*.

Moses S. Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM Symposium on Theory of Computing*, pages 380–388.

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, Series B*, 39(1):1–38.

Qing Dou and Kevin Knight. 2012. Large scale decipherment for out-of-domain machine translation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 266–275.

Pascale Fung and Kathleen McKeown. 1997. Finding terminology translations from non-parallel corpora. In *Proceedings of the 5th Annual Workshop on Very Large Corpora*, pages 192–202.

Sharon Goldwater and Tom Griffiths. 2007. A fully bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 744–751.

Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL: HLT*, pages 771–779.

Mark Johnson. 2007. Why doesn't EM find good HMM POS-taggers? In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 296–305.

Alex Klementiev, Ann Irvine, Chris Callison-Burch, and David Yarowsky. 2012. Toward statistical machine translation without parallel corpora. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*.

Kevin Knight and Kenji Yamada. 1999. A computational approach to deciphering unknown scripts. In *Proceedings of the ACL Workshop on Unsupervised Learning in Natural Language Processing*, pages 37–44.

Philipp Koehn and Kevin Knight. 2000. Estimating word translation probabilities from unrelated monolingual corpora using the em algorithm. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 711–715.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180.

Malte Nuhn, Arne Mauser, and Hermann Ney. 2012. Deciphering foreign language by combining language models and context vectors. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 156–164.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318.

Reinhard Rapp. 1995. Identifying word translations in non-parallel texts. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, pages 320–322.

Sujith Ravi and Kevin Knight. 2011a. Bayesian inference for zodiac and other homophonic ciphers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 239–247.

Sujith Ravi and Kevin Knight. 2011b. Deciphering foreign language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 12–21.

Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. 2005. Randomized algorithms and nlp: using locality sensitive hash function for high speed noun clustering. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 622–629.

Benjamin Snyder, Regina Barzilay, and Kevin Knight. 2010. A statistical model for lost language decipherment. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1048–1057.

Jörg Tiedemann. 2009. News from opus - a collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248.