Beyond Parallel Data: Joint Word Alignment and Decipherment Improves Machine Translation

Qing Dou , Ashish Vaswani, and Kevin Knight
Information Sciences Institute
Department of Computer Science
University of Southern California
{qdou,avaswani,knight}@isi.edu

Abstract

Inspired by previous work, where decipherment is used to improve machine translation, we propose a new idea to combine word alignment and decipherment into a single learning process. We use EM to estimate the model parameters, not only to maximize the probability of parallel corpus, but also the monolingual corpus. We apply our approach to improve Malagasy-English machine translation, where only a small amount of parallel data is available. In our experiments, we observe gains of 0.9 to 2.1 BLEU over a strong baseline.

1 Introduction

State-of-the-art machine translation (MT) systems apply statistical techniques to learn translation rules automatically from parallel data. However, this reliance on parallel data seriously limits the scope of MT application in the real world, as for many languages and domains, there is not enough parallel data to train a decent quality MT system.

However, compared with parallel data, there are much larger amounts of non parallel data. The ability to learn a translation lexicon or even build a machine translation system using monolingual data helps address the problems of insufficient parallel data. Ravi and Knight (2011) are among the first to learn a full MT system using only non parallel data through decipherment. However, the performance of such systems is much lower compared with those trained with parallel data. In another work, Klementiev et al. (2012) show that, given a phrase table, it is possible to estimate parameters for a phrase-based MT system from non parallel data.

Given that we often have some parallel data, it is more practical to improve a translation system trained on parallel data by using additional non parallel data. Rapp (1995) shows that with a seed lexicon, it is possible to induce new word level translations from non parallel data. Motivated by the idea that a translation lexicon induced from non parallel data can be used to translate out of vocabulary words (OOV), a variety of prior research has tried to build a translation lexicon from non parallel or comparable data (Fung and Yee, 1998; Koehn and Knight, 2002; Haghighi et al., 2008; Garera et al., 2009; Bergsma and Van Durme, 2011; Daumé and Jagarlamudi, 2011; Irvine and Callison-Burch, 2013b; Irvine and Callison-Burch, 2013a; Irvine et al., 2013).

Lately, there has been increasing interest in learning translation lexicons from non parallel data with decipherment techniques (Ravi and Knight, 2011; Dou and Knight, 2012; Nuhn et al., 2012; Dou and Knight, 2013). Decipherment views one language as a cipher for another and learns a translation lexicon that produces fluent text in the target (plaintext) language. Previous work has shown that decipherment not only helps find translations for OOVs (Dou and Knight, 2012), but also improves translations of observed words (Dou and Knight, 2013).

We find that previous work using monolingual or comparable data to improve quality of machine translation separates two learning tasks: first, translation rules are learned from parallel data, and then the information learned from parallel data is used to bootstrap learning with non parallel data. Inspired by approaches where joint inference reduces the problems of error propagation and improves system performance, we combine the two separate learning processes into a single one, as shown in Figure 1. The contributions of this work are:

- We propose a new objective function for word alignment that combines the process of word alignment and decipherment into a single learning task.

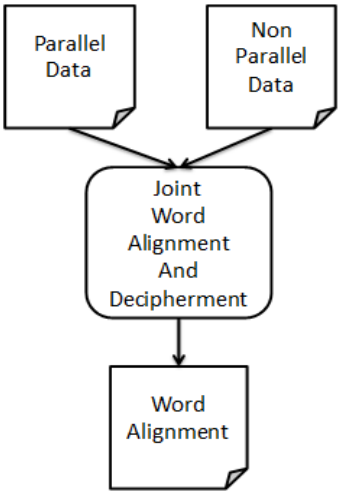

Figure 1: Combine word alignment and decipherment into a single learning process.

- In experiments, we find that the joint process outperforms the previous pipeline approach, and observe BLEU gains of 0.9 and 2.1 on two different test sets.
- We release 15.3 million tokens of monolingual Malagasy data from the web, as well as a small Malagasy dependency tree bank containing 20k tokens.

2 Joint Word Alignment and Decipherment

2.1 A New Objective Function

In previous work that uses monolingual data to improve machine translation, a seed translation lexicon learned from parallel data is used to find new translations through either word vector based approaches or decipherment. In return, selection of a seed lexicon needs to be careful as using a poor quality seed lexicon could hurt the downstream process. Evidence from a number of previous work shows that a joint inference process leads to better performance in both tasks (Jiang et al., 2008; Zhang and Clark, 2008).

In the presence of parallel and monolingual data, we would like the alignment and decipherment models to benefit from each other. Since the decipherment and word alignment models contain word-to-word translation probabilities $t(f \mid e)$, having them share these parameters during learning will allow us to pool information from both data types. This leads us to develop a new objective function that takes both learning processes into account. Given our parallel data, $(\mathbf{E}^1, \mathbf{F}^1), \ldots, (\mathbf{E}^m, \mathbf{F}^m), \ldots, (\mathbf{E}^M, \mathbf{F}^M)$, and monolingual data $\mathbf{F}^1_{\mathbf{mono}}, \ldots, \mathbf{F}^n_{\mathbf{mono}}, \ldots, \mathbf{F}^N_{\mathbf{mono}}$, we seek to maximize the likelihood of both. Our new objective function is defined as:

$$F_{joint} = \sum_{m=1}^{M} \log P(\mathbf{F}^m \mid \mathbf{E}^m) + \alpha \sum_{n=1}^{N} \log P(\mathbf{F}^n_{\mathbf{mono}}) \quad (1)$$

The goal of training is to learn the parameters that maximize this objective, that is

$$\theta^* = \arg\max_{\theta} F_{joint} \quad (2)$$

In the next two sections, we describe the word alignment and decipherment models, and present how they are combined to perform joint optimization.

2.2 Word Alignment

Given a source sentence $\mathbf{F} = \mathbf{f_1}, \ldots, \mathbf{f_j}, \ldots, \mathbf{f_J}$ and a target sentence $\mathbf{E} = \mathbf{e_1}, \ldots, \mathbf{e_i}, \ldots, \mathbf{e_I}$, word alignment models describe the generative process employed to produce the French sentence from the English sentence through alignments $\mathbf{a} = \mathbf{a_1}, \ldots, \mathbf{a_j}, \ldots, \mathbf{a_J}$.

The IBM models 1-2 (Brown et al., 1993) and the HMM word alignment model (Vogel et al., 1996) use two sets of parameters, *distortion* probabilities and *translation* probabilities, to define the joint probability of a target sentence and alignment given a source sentence.

$$P(\mathbf{F}, \mathbf{a} \mid \mathbf{E}) = \prod_{j=1}^{J} d(a_j \mid a_{j-1}, j) t(f_j \mid e_{a_j}). \quad (3)$$

These alignment models share the same translation probabilities $t(f_j \mid e_{a_j})$, but differ in their treatment of the distortion probabilities $d(a_j \mid a_{j-1}, j)$. Brown et al. (1993) introduce more advanced models for word alignment, such as Model 3 and Model 4, which use more parameters to describe the generative process. We do not go into details of those models here and the reader is referred to the paper describing them.

Under the Model 1-2 and HMM alignment models, the probability of target sentence given source sentence is:

$$P(\mathbf{F} \mid \mathbf{E}) = \sum_{\mathbf{a}} \prod_{j=1}^{J} d(a_j \mid a_{j-1}, j) t(f_j \mid e_{a_j}).$$

Let θ denote all the parameters of the word alignment model. Given a corpus of sentence pairs $(\mathbf{E}^1, \mathbf{F}^1), \ldots, (\mathbf{E}^m, \mathbf{F}^m), \ldots, (\mathbf{E}^M, \mathbf{F}^M)$, the standard approach for training is to learn the maximum likelihood estimate of the parameters, that is,

$$\begin{aligned}\theta^* &= \arg\max_{\theta} \sum_{m=1}^{M} \log P(\mathbf{F}^m \mid \mathbf{E}^m) \\ &= \arg\max_{\theta} \log \left(\sum_{\mathbf{a}} P(\mathbf{F}^m, \mathbf{a} \mid \mathbf{E}^m) \right).\end{aligned}$$

We typically use the EM algorithm (Dempster et al., 1977), to carry out this optimization.

2.3 Decipherment

Given a corpus of N foreign text sequences (ciphertext), $\mathbf{F}^1_{\mathbf{mono}}, \ldots, \mathbf{F}^n_{\mathbf{mono}}, \ldots, \mathbf{F}^N_{\mathbf{mono}}$, decipherment finds word-to-word translations that best describe the ciphertext.

Knight et al. (2006) are the first to study several natural language decipherment problems with unsupervised learning. Since then, there has been increasing interest in improving decipherment techniques and its application to machine translation (Ravi and Knight, 2011; Dou and Knight, 2012; Nuhn et al., 2012; Dou and Knight, 2013; Nuhn et al., 2013).

In order to speed up decipherment, Dou and Knight (2012) suggest that a frequency list of bigrams might contain enough information for decipherment. According to them, a monolingual ciphertext bigram $\mathbf{F}_{\mathbf{mono}}$ is generated through the following generative story:

- Generate a sequence of two plaintext tokens $e_1 e_2$ with probability $P(e_1 e_2)$ given by a language model built from large numbers of plaintext bigrams.
- Substitute e_1 with f_1 and e_2 with f_2 with probability $t(f_1|e_1) \cdot t(f_2|e_2)$.

The probability of any cipher bigram F is:

$$P(\mathbf{F}_{\mathbf{mono}}) = \sum_{e_1 e_2} P(e_1 e_2) \cdot t(f_1|e_1) \cdot t(f_2|e_2) \quad (4)$$

And the probability of the corpus is:

$$P(corpus) = \prod_{n=1}^{N} P(\mathbf{F}^n_{\mathbf{mono}}) \quad (5)$$

Given a plaintext bigram language model, the goal is to manipulate $t(f|e)$ to maximize $P(corpus)$. Theoretically, one can directly apply EM to solve the problem (Knight et al., 2006). However, EM has time complexity $O(N \cdot V_e^2)$ and space complexity $O(V_f \cdot V_e)$, where V_f, V_e are the sizes of ciphertext and plaintext vocabularies respectively, and N is the number of cipher bigrams.

There have been previous attempts to make decipherment faster. Ravi and Knight (2011) apply Bayesian learning to reduce the space complexity. However, Bayesian decipherment is still very slow with Gibbs sampling (Geman and Geman, 1987). Dou and Knight (2012) make sampling faster by introducing slice sampling (Neal, 2000) to Bayesian decipherment. Besides Bayesian decipherment, Nuhn et al. (2013) show that beam search can be used to solve a very large 1:1 word substitution cipher. In subsection 2.4.1, we describe our approach that uses slice sampling to compute expected counts for decipherment in the EM algorithm.

2.4 Joint Optimization

We now describe our EM approach to learn the parameters that maximize F_{joint} (equation 2), where the distortion probabilities, $d(a_j \mid a_{j-1}, j)$ in the word alignment model are only learned from parallel data, and the translation probabilities, $t(f \mid e)$ are learned using both parallel and non parallel data. The E step and M step are illustrated in Figure 2.

Our algorithm starts with EM learning only on parallel data for a few iterations. When the joint inference starts, we first compute expected counts from parallel data and non parallel data using parameter values from the last M step separately. Then, we add the expected counts from both parallel data and non parallel data together with different weights for the two. Finally we renormalize the translation table and distortion table to update parameters in the new M step.

The E step for parallel part can be computed efficiently using the forward-backward algorithm (Vogel et al., 1996). However, as we pointed out in Section 2.3, the E step for the non parallel part has a time complexity of $O(V^2)$ with the forward-backward algorithm, where V is the size of English vocabulary, and is usually very large. Previous work has tried to make decipherment scalable (Ravi and Knight, 2011; Dou and Knight, 2012;

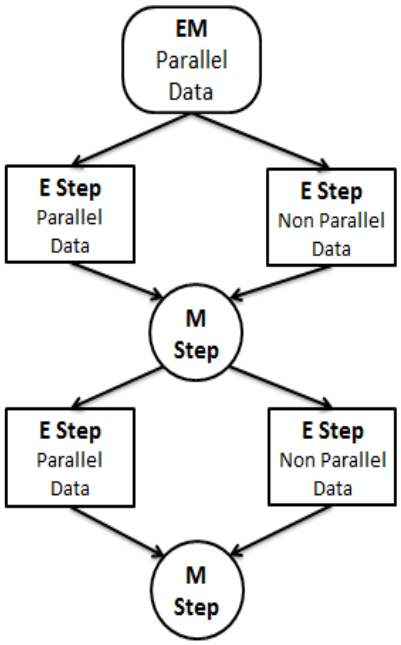

Figure 2: Joint Word Alignment and Decipherment with EM

Nuhn et al., 2013; Ravi, 2013). However, all of them are designed for decipherment with either Bayesian inference or beam search. In contrast, we need an algorithm to make EM decipherment scalable. To overcome this problem, we modify the slice sampling (Neal, 2000) approach used by Dou and Knight (2012) to compute expected counts from non parallel data needed for the EM algorithm.

2.4.1 Draw Samples with Slice Sampling

To start the sampling process, we initialize the first sample by performing approximate Viterbi decoding using results from the last EM iteration. For each foreign dependency bigram f_1, f_2, we find the top 50 candidates for f_1 and f_2 ranked by $t(e|f)$, and find the English sequence e_1, e_2 that maximizes $t(e_1|f_1) \cdot t(e_2|f_2) \cdot P(e_1, e_2)$.

Suppose the derivation probability for current sample *e_current* is $P(e\_current)$, we use slice sampling to draw a new sample in two steps:

- Select a threshold T uniformly between 0 and $P(e\_current)$.
- Draw a new sample *e_new* uniformly from a pool of candidates: $\{e\_new|P(e\_new) > T\}$.

The first step is straightforward to implement. However, it is not trivial to implement the second step. We adapt the idea from Dou and Knight (2012) for EM learning.

Suppose our current sample *e_current* contains English tokens e_{i-1}, e_i, and e_{i+1} at position $i-1$, i, and $i+1$ respectively, and f_i be the foreign token at position i. Using point-wise sampling, we draw a new sample by changing token e_i to a new token e'. Since the rest of the sample remains the same, only the probability of the trigram $P(e_{i-1}e'e_{i+1})$ (The probability is given by a bigram language model.), and the channel model probability $t(f_i|e')$ change. Therefore, the probability of a sample is simplified as shown Equation 6.

$$P(e_{i-1}e'e_{i+1}) \cdot t(f_i|e') \tag{6}$$

Remember that in slice sampling, a new sample is drawn in two steps. For the first step, we choose a threshold T uniformly between 0 and $P(e_{i-1}e_ie_{i+1}) \cdot t(f_i|e_i)$. We divide the second step into two cases based on the observation that two types of samples are more likely to have a probability higher than T (Dou and Knight, 2012): (1) those whose trigram probability is high, and (2) those whose channel model probability is high. To find candidates that have high trigram probability, Dou and Knight (2012) build a top k sorted lists ranked by $P(e_{i-1}e'e_{i+1})$, which can be precomputed off-line. Then, they test if the last item e_k in the list satisfies the following inequality:

$$P(e_{i-1}e_ke_{i+1}) \cdot c < T \tag{7}$$

where c is a small constant and is set to *prior* in their work. In contrast, we choose c empirically as we do not have a prior in our model. When the inequality in Equation 7 is satisfied, a sample is drawn in the following way: Let set $A = \{e'|e_{i-1}e'e_{i+1} \cdot c > T\}$ and set $B = \{e'|t(f_i|e') > c\}$. Then we only need to sample e' uniformly from $A \cup B$ until $P(e_{i-1}e'e_{i+1}) \cdot t(f_i|e')$ is greater than T. It is easy to prove that all other candidates that are not in the sorted list and with $t(f_i|e') \leq c$ have a upper bound probability: $P(e_{i-1}e_ke_{i+1}) \cdot c$. Therefore, they do not need to be considered.

Second, when the last item e_k in the list does not meet the condition in Equation 7, we keep drawing samples e' randomly until its probability is greater than the threshold T.

As we mentioned before, the choice of the small constant c is empirical. A large c reduces the number of items in set B, but makes the condition $P(e_{i-1}e_ke_{i+1}) \cdot c < T$ less likely to satisfy, which

slows down the sampling. On the contrary, a small c increases the number of items in set B significantly as EM does not encourage a sparse distribution, which also slows down the sampling. In our experiments, we set c to 0.001 based on the speed of decipherment. Furthermore, to reduce the size of set B, we rank all the candidate translations of f_i by $t(e'|f_i)$, then we add maximum the first 1000 candidates whose $t(f_i|e') >= c$ into set B. For the rest of the candidates, we set $t(f_i|e')$ to a value smaller than c (0.00001 in experiments).

2.4.2 Compute Expected Counts from Samples

With the ability to draw samples efficiently for decipherment using EM, we now describe how to compute expected counts from those samples. Let f_1, f_2 be a specific ciphertext bigram, N be the number of samples we want to use to compute expected counts, and e_1, e_2 be one of the N samples. The expected counts for pairs (f_1, e_1) and (f_2, e_2) are computed as:

$$\alpha \cdot \frac{count(f_1, f_2)}{N}$$

where $count(f_1, f_2)$ is count of the bigram, and α is the weight for non parallel data as shown in Equation 1. Expected counts collected for f_1, f_2 are accumulated from each of its N samples. Finally, we collect expected counts using the same approach from each foreign bigram.

3 Word Alignment Experiments

In this section, we show that joint word alignment and decipherment improves the quality of word alignment. We choose to evaluate word alignment performance for Spanish and English as manual gold alignments are available. In experiments, our approach improves alignment F score by as much as 8 points.

3.1 Experiment Setup

As shown in Table 1, we work with a small amount of parallel, manually aligned Spanish-English data (Lambert et al., 2005), and a much larger amount of monolingual data.

The parallel data is extracted from Europarl, which consists of articles from European parliament plenary sessions. The monolingual data comes from English and Spanish versions of Gigaword corpra containing news articles from different news agencies.

	Spanish	English
Parallel	10.3k	9.9k
Non Parallel	80 million	400 million

Table 1: Size of parallel and non parallel data for word alignment experiments (Measured in number of tokens)

We view Spanish as a cipher of English, and follow the approach proposed by Dou and Knight (2013) to extract dependency bigrams from parsed Spanish and English monolingual data for decipherment. We only keep bigrams where both tokens appear in the parallel data. Then, we perform Spanish to English (English generating Spanish) word alignment and Spanish to English decipherment simultaneously with the method discussed in section 2.

3.1.1 Results

We align all 500 sentences in the parallel corpus, and tune the decipherment weight (α) for Model 1 and HMM using the last 100 sentences. The best weights are 0.1 for Model 1, and 0.005 for HMM. We start with Model 1 with only parallel data for 5 iterations, and switch to the joint process for another 5 iterations with Model 1 and 5 more iterations of HMM. In the end, we use the first 100 sentence pairs of the corpus for evaluation.

Figure 3 compares the learning curve of alignment F-score between EM without decipherment (baseline) and our joint word alignment and decipherment. From the learning curve, we find that at the 6th iteration, 2 iterations after we start the joint process, alignment F-score is improved from 34 to 43, and this improvement is held through the rest of the Model 1 iterations. The alignment model switches to HMM from the 11th iteration, and at the 12th iteration, we see a sudden jump in F-score for both the baseline and the joint approach. We see consistent improvement of F-score till the end of HMM iterations.

4 Improving Low Density Languages Machine Translation with Joint Word Alignment and Decipherment

In the previous section, we show that the joint word alignment and decipherment process improves quality of word alignment significantly for Spanish and English. In this section, we test our approach in a more challenging setting: improv-

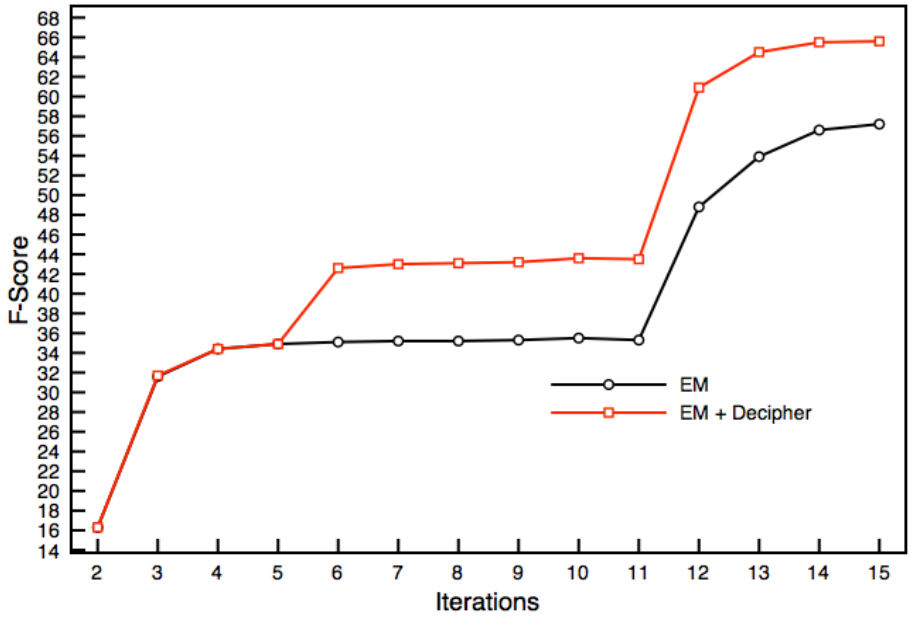

Figure 3: Learning curve showing our joint word alignment and decipherment approach improves word alignment quality over the traditional EM without decipherment (Model 1: Iteration 1 to 10, HMM: Iteration 11 to 15)

ing the quality of machine translation in a real low density language setting.

In this task, our goal is to build a system to translate Malagasy news into English. We have a small amount of parallel data, and larger amounts of monolingual data collected from online websites. We build a dependency parser for Malagasy to parse the monolingual data to perform dependency based decipherment (Dou and Knight, 2013). In the end, we perform joint word alignment and decipherment, and show that the joint learning process improves BLEU scores by up to 2.1 points over a phrase-based MT baseline.

4.1 The Malagasy Language

Malagasy is the official language of Madagascar. It has around 18 million native speakers. Although Madagascar is an African country, Malagasy belongs to the Malayo-Polynesian branch of the Austronesian language family. Malagasy and English have very different word orders. First of all, in contrast to English, which has a subject-verb-object (SVO) word order, Malagasy has a verb-object-subject (VOS) word order. Besides that, Malagasy is a typical head initial language: Determiners precede nouns, while other modifiers and relative clauses follow nouns (e.g. ny "the" boky "book" mena "red"). The significant differences in word order pose great challenges for both machine translation and decipherment.

4.2 Data

Table 2 shows the data available to us in our experiments. The majority of parallel text comes from Global Voices[1] (GV). The website contains international news translated into different foreign languages. Besides that, we also have a very small amount of parallel text containing local web news, with English translations provided by native speakers at the University of Texas, Austin. The Malagasy side of this small parallel corpus also has syntactical annotation, which is used to train a very basic Malagasy part of speech tagger and dependency parser.

We also have much larger amounts of non parallel data for both languages. For Malagasy, we spent two months manually collecting 15.3 million tokens of news text from local news websites in Madagascar.[2] We have released this data for future research use. For English, we have 2.4 billion tokens from the Gigaword corpus. Since the Malagasy monolingual data is collected from local websites, it is reasonable to argue that those

[1] globalvoicesonline.org

[2] aoraha.com, gazetiko.com, inovaovao.com, expressmada.com, lakroa.com

Source	Malagasy	English
Parallel		
Global Voices	2.0 million	1.8 million
Web News	2.2k	2.1k
Non Parallel		
Gigaword	N/A	2.4 billion
allAfrica	N/A	396 million
Local News	15.3 million	N/A

Table 2: Size of Malagasy and English data used in our experiments (Measured in number of tokens)

data contain significant amount of information related to Africa. Therefore, we also collect 396 million tokens of African news in English from allAfrica.com.

4.3 Building A Dependency Parser for Malagasy

Since Malagasy and English have very different word orders, we decide to apply dependency based decipherment for the two languages as suggested by Dou and Knight (2013). To extract dependency relations, we need to parse monolingual data in Malagasy and English. For English, there are already many good parsers available. In our experiments, we use Turbo parser (Martins et al., 2013) trained on the English Penn Treebank (Marcus et al., 1993) to parse all our English monolingual data. However, there is no existing good parser for Malagasy.

The quality of a dependency parser depends on the amount of training data available. State-of-the-art English parsers are built from Penn Treebank, which contains over 1 million tokens of annotated syntactical trees. In contrast, the available data for training a Malagasy parser is rather limited, with only 168 sentences, and 2.8k tokens, as shown in Table 2. At the very beginning, we use the last 120 sentences as training data to train a part of speech (POS) tagger using a toolkit provided by Garrette et al. (2013) and a dependency parser with the Turbo parser. We test the performance of the parser on the first 48 sentences and obtain 72.4% accuracy.

One obvious way to improve tagging and parsing accuracy is to get more annotated data. We find more data with only part of speech tags containing 465 sentences and 10k tokens released by (Garrette et al., 2013), and add this data as extra training data for POS tagger. Also, we download an online dictionary that contains POS tags for over 60k Malagasy word types from malagasyword.org. The dictionary is very helpful for tagging words never seen in the training data.

It is natural to think that creation of annotated data for training a POS tagger and a parser requires large amounts of efforts from annotators who understand the language well. However, we find that through the help of parallel data and dictionaries, we are able to create more annotated data by ourselves to improve tagging and parsing accuracy. This idea is inspired by previous work that tries to learn a semi-supervised parser by projecting dependency relations from one language (with good dependency parsers) to another (Yarowsky and Ngai, 2001; Ganchev et al., 2009). However, we find those automatic approaches do not work well for Malagasy.

To further expand our Malagasy training data, we first use a POS tagger and parser with poor performance to parse 788 sentences (20k tokens) on the Malagasy side of the parallel corpus from Global Voices. Then, we correct both the dependency links and POS tags based on information from dictionaries[3] and the English translation of the parsed sentence. We spent 3 months to manually project English dependencies to Malagasy and eventually improve test set parsing accuracy from 72.4% to 80.0%. We also make this data available for future research use.

4.4 Machine Translation Experiments

In this section, we present the data used for our MT experiments, and compare three different systems to justify our joint word alignment and decipherment approach.

4.4.1 Baseline Machine Translation System

We build a state-of-the-art phrase-based MT system, PBMT, using Moses (Koehn et al., 2007). PBMT has 3 models: a translation model, a distortion model, and a language model. We train the other models using half of the Global Voices parallel data (the rest is reserved for development and testing), and build a 5-gram language model using 834 million tokens from AFP section of English Gigaword, 396 million tokens from allAfrica, and the English part of the parallel corpus for training. For alignment, we run 10 iterations of Model 1, and 5 iterations of HMM. We did not run Model 3

[3] an online dictionary from malagasyword.org, as well as a lexicon learned from the parallel data

and Model 4 as we see no improvements in BLEU scores from running those models. We do word alignment in two directions and use grow-diag-final-and heuristic to obtain final alignment. During decoding, we use 8 standard features in Moses to score a candidate translation: direct and inverse translation probabilities, direct and inverse lexical weighting, a language model score, a distortion score, phrase penalty, and word penalty. The weights for the features are learned on the tuning data using minimum error rate training (MERT) (Och, 2003).

To compare with previous decipherment approach to improve machine translation, we build a second baseline system. We follow the work by Dou and Knight (2013) to decipher Malagasy into English, and build a translation lexicon $T_{decipher}$ from decipherment. To improve machine translation, we simply use $T_{decipher}$ as an additional parallel corpus. First, we filter $T_{decipher}$ by keeping only translation pairs (f, e), where f is observed in the Spanish part and e is observed in the English part of the parallel corpus. Then we append all the Spanish and English words in the filtered $T_{decipher}$ to the end of Spanish part and English part of the parallel corpus respectively. The training and tuning process is the same as the baseline machine translation system PBMT. We call this system **Decipher-Pipeline**.

4.4.2 Joint Word Alignment and Decipherment for Machine Translation

When deciphering Malagasy to English, we extract Malagasy dependency bigrams using all available Malagasy monolingual data plus the Malagasy part of the Global Voices parallel data, and extract English dependency bigrams using 834 million tokens from English Gigaword, and 396 million tokens from allAfrica news to build an English dependency language model. In the other direction, we extract English dependency bigrams from English part of the entire parallel corpus plus 9.7 million tokens from allAfrica news [4], and use 17.3 million tokens Malagasy monolingual data (15.3 million from the web and 2.0 million from Global Voices) to build a Malagasy dependency language model. We require that all dependency bigrams only contain words observed in the parallel data used to train the baseline MT system.

[4]We do not find further BLEU gains by using more English monolingual data.

Parallel		
	Malagasy	English
Train(GV)	0.9 million	0.8 million
Tune(GV)	22.2k	20.2k
Test(GV)	23k	21k
Test(Web)	2.2k	2.1k
Non Parallel		
	Malagasy	English
Gigaword	N/A	834 million
Web	15.3 million	396 million

Table 3: Size of training, tuning, and testing data in number of tokens (GV:Global Voices)

During learning, we run Model 1 without decipherment for 5 iterations. Then we perform joint word alignment and decipherment for another 5 iterations with Model 1 and 5 iterations with HMM. We tune decipherment weights (α) for Model 1 and HMM using grid search against BLEU score on a development set. In the end, we only extract rules from one direction $P(English|Malagasy)$, where the decipherment weights for Model 1 and HMM are 0.5 and 0.005 respectively. We chose this because we did not find any benefits to tune the weights on each direction, and then use grow-diag-final-end heuristic to form final alignments. We call this system **Decipher-Joint**.

4.5 Results

We tune each system three times with MERT and choose the best weights based on BLEU scores on tuning set.

Table 4 shows that while using a translation lexicon learnt from decipherment does not improve the quality of machine translation significantly, the joint approach improves BLEU score by 0.9 and 2.1 on Global Voices test set and web news test set respectively. The results show that the parsing quality correlates with gains in BLEU scores. Scores in the brackets in the last row of the table are achieved using a dependency parser with 72.4% attachment accuracy, while scores outside the brackets are obtained using a dependency parser with 80.0% attachment accuracy.

We analyze the results and find the gain mainly comes from two parts. First, adding expected counts from non parallel data makes the distribution of translation probabilities sparser in word alignment models. The probabilities of translation pairs favored by both parallel data and decipher-

Decipherment	System	Tune (GV)	Test (GV)	Test (Web)
None	PBMT (Baseline)	18.5	17.1	7.7
Separate	Decipher-Pipeline	18.5	17.4	7.7
Joint	Decipher-Joint	18.9 (18.7)	18.0 (17.7)	9.8 (8.5)

Table 4: Decipher-Pipeline does not show significant improvement over the baseline system. In contrast, Decipher-Joint using joint word alignment and decipherment approach achieves a BLEU gain of 0.9 and 2.1 on the Global Voices test set and the web news test set, respectively. The results in brackets are obtained using a parser trained with only 120 sentences. (GV : Global Voices)

ment becomes higher. This gain is consistent with previous observation where a sparse prior is applied to EM to help improve word alignment and machine translation (Vaswani et al., 2012). Second, expected counts from decipherment also help discover new translation pairs in the parallel data for low frequency words, where those words are either aligned to NULL or wrong translations in the baseline.

5 Conclusion and Future Work

We propose a new objective function for word alignment to combine the process of word alignment and decipherment into a single task. In, experiments, we find that the joint process performs better than previous pipeline approach, and observe BLEU gains of 0.9 and 2.1 point on Global Voices and local web news test sets, respectively. Finally, our research leads to the release of 15.3 million tokens of monolingual Malagasy data from the web as well as a small Malagasy dependency tree bank containing 20k tokens.

Given the positive results we obtain by using the joint approach to improve word alignment, we are inspired to apply this approach to help find translations for out of vocabulary words, and to explore other possible ways to improve machine translation with decipherment.

6 Acknowledgments

We would like to thank David Chiang, Tomer Levinboim, Nima Pourdamghani, Theerawat Songyot, Allen Schmaltz, and Julian Schamper for their feedback. This work was supported by ARL/ARO (W911NF-10-1-0533) and DARPA (HR0011-12-C-0014).

References

Shane Bergsma and Benjamin Van Durme. 2011. Learning bilingual lexicons using the visual similarity of labeled web images. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Three*. AAAI Press.

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*, 19:263–311.

Hal Daumé, III and Jagadeesh Jagarlamudi. 2011. Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Arthur Dempster, Nan Laird, and Donald Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Computational Linguistics*, 39(4):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*. Association for Computational Linguistics.

Qing Dou and Kevin Knight. 2013. Dependency-based decipherment for resource-limited machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Pascale Fung and Lo Yuen Yee. 1998. An IR approach for translating new words from nonparallel, comparable texts. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*. Association for Computational Linguistics.

Kuzman Ganchev, Jennifer Gillenwater, and Ben Taskar. 2009. Dependency grammar induction via bitext projection constraints. 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: Volume 1 - Volume 1*. Association for Computational Linguistics.

Nikesh Garera, Chris Callison-Burch, and David Yarowsky. 2009. Improving translation lexicon induction from monolingual corpora via dependency contexts and part-of-speech equivalences. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics.

Dan Garrette, Jason Mielens, and Jason Baldridge. 2013. Real-world semi-supervised learning of pos-taggers for low-resource languages. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.

Stuart Geman and Donald Geman. 1987. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. In *Readings in computer vision: issues, problems, principles, and paradigms*. Morgan Kaufmann Publishers Inc.

Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL-08: HLT*. Association for Computational Linguistics.

Ann Irvine and Chris Callison-Burch. 2013a. Combining bilingual and comparable corpora for low resource machine translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, August.

Ann Irvine and Chris Callison-Burch. 2013b. Supervised bilingual lexicon induction with multiple monolingual signals. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Ann Irvine, Chris Quirk, and Hal Daume III. 2013. Monolingual marginal matching for translation model adaptation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Wenbin Jiang, Liang Huang, Qun Liu, and Yajuan Lü. 2008. A cascaded linear model for joint Chinese word segmentation and part-of-speech tagging. In *Proceedings of ACL-08: HLT*. Association for Computational Linguistics.

Alexandre 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*. Association for Computational Linguistics.

Kevin Knight, Anish Nair, Nishit Rathod, and Kenji Yamada. 2006. Unsupervised analysis for decipherment problems. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*. Association for Computational Linguistics.

Philipp Koehn and Kevin Knight. 2002. Learning a translation lexicon from monolingual corpora. In *Proceedings of the ACL-02 Workshop on Unsupervised Lexical Acquisition*. Association for Computational Linguistics.

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*. Association for Computational Linguistics.

Patrik Lambert, Adriá De Gispert, Rafael Banchs, and José B. Mariño. 2005. Guidelines for word alignment evaluation and manual alignment. *Language Resources and Evaluation*, 39(4):267–285.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2).

Andre Martins, Miguel Almeida, and Noah A. Smith. 2013. Turning on the Turbo: Fast third-order non-projective Turbo parsers. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics.

Radford Neal. 2000. Slice sampling. *Annals of Statistics*, 31.

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: Long Papers - Volume 1*. Association for Computational Linguistics.

Malte Nuhn, Julian Schamper, and Hermann Ney. 2013. Beam search for solving substitution ciphers. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of Association for Computational Linguistics*. Association for Computational Linguistics.

Reinhard Rapp. 1995. Identifying word translations in non-parallel texts. In *Proceedings of the 33rd annual meeting of Association for Computational Linguistics*. Association for Computational Linguistics.

Sujith Ravi and Kevin Knight. 2011. Deciphering foreign language. In *Proceedings of the 49th Annual*

Meeting of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics.

Sujith Ravi. 2013. Scalable decipherment for machine translation via hash sampling. In *Proceedings of the 51th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Ashish Vaswani, Liang Huang, and David Chiang. 2012. Smaller alignment models for better translations: Unsupervised word alignment with the l0-norm. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*. Association for Computational Linguistics.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 2*. Association for Computational Linguistics.

David Yarowsky and Grace Ngai. 2001. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies*. Association for Computational Linguistics.

Yue Zhang and Stephen Clark. 2008. Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of ACL-08: HLT*, Columbus, Ohio. Association for Computational Linguistics.