# Dependency-Based Decipherment for Resource-Limited Machine Translation

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

## Abstract

We introduce dependency relations into deciphering foreign languages and show that dependency relations help improve the state-of-the-art deciphering accuracy by over 500%. We learn a translation lexicon from large amounts of genuinely non parallel data with decipherment to improve a phrase-based machine translation system trained with limited parallel data. In experiments, we observe BLEU gains of 1.2 to 1.8 across three different test sets.

## 1 Introduction

State-of-the-art machine translation (MT) systems apply statistical techniques to learn translation rules from large amounts of parallel data. However, parallel data is limited for many language pairs and domains.

In general, it is easier to obtain non parallel data. The ability to build a machine translation system using monolingual data could alleviate problems caused by insufficient parallel data. Towards building a machine translation system without a parallel corpus, Klementiev et al. (2012) use non parallel data to estimate parameters for a large scale MT system. Other work tries to learn full MT systems using only non parallel data through decipherment (Ravi and Knight, 2011; Ravi, 2013). However, the performance of such systems is poor compared with those trained with parallel data.

Given that we often have some parallel data, it is more practical to improve a translation system trained on parallel corpora with non parallel data. Dou and Knight (2012) successfully apply decipherment to learn a domain specific translation lexicon from monolingual data to improve out-of-domain machine translation. Although their approach works well for Spanish/French, they do not show whether their approach works for other language pairs. Moreover, the non parallel data used in their experiments is created from a parallel corpus. Such highly comparable data is difficult to obtain in reality.

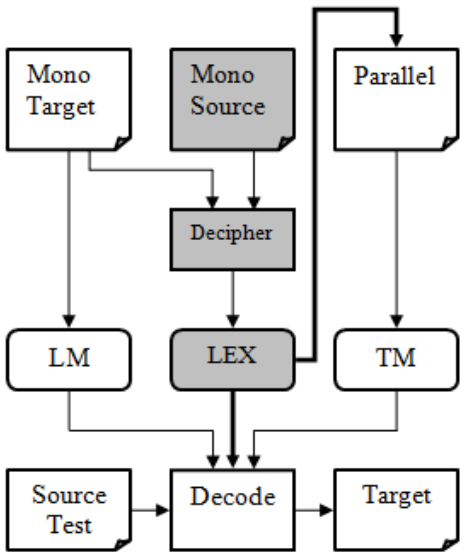


Figure 1: Improving machine translation with decipherment (Grey boxes represent new data and process). Mono: monolingual; LM: language model; LEX: translation lexicon; TM: translation model.

In this work, we improve previous work by Dou and Knight (2012) using genuinely non parallel data,



*Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1668–1676,
Seattle, Washington, USA, 18-21 October 2013. 

and propose a framework to improve a machine translation system trained with a small amount of parallel data. As shown in Figure 1, we use a lexicon learned from decipherment to improve translations of both observed and out-of-vocabulary (OOV) words. The main contributions of this work are:

- We extract bigrams based on dependency relations for decipherment, which improves the state-of-the-art deciphering accuracy by over 500%.
- We demonstrate how to improve translations of words observed in parallel data by using a translation lexicon obtained from large amounts of non parallel data.
- We show that decipherment is able to find correct translations for OOV words.
- We use a translation lexicon learned by deciphering large amounts of non parallel data to improve a phrase-based MT system trained with limited amounts of parallel data. In experiments, we observe 1.2 to 1.8 BLEU gains across three different test sets.

## 2 Previous Work

Motivated by the idea that a translation lexicon induced from non parallel data can be applied to MT, a variety of prior research has tried to build a translation lexicon from non parallel or comparable data (Rapp, 1995; 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). Although previous work is able to build a translation lexicon without parallel data, little has used the lexicon to improve machine translation.

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). Decipherment views one language as a cipher for another and learns a translation lexicon that produces a good decipherment.

In an effort to build a MT system without a parallel corpus, Ravi and Knight (2011) view Spanish as a cipher for English and apply Bayesian learning to directly decipher Spanish into English. Unfortunately, their approach can only work on small data with limited vocabulary. Dou and Knight (2012) propose two techniques to make Bayesian decipherment scalable.

First, unlike Ravi and Knight (2011), who decipher whole sentences, Dou and Knight (2012) decipher bigrams. Reducing a ciphertext to a set of bigrams with counts significantly reduces the amount of cipher data. According to Dou and Knight (2012), a ciphertext bigram $F$ is generated through the following generative story:

- Generate a sequence of two plaintext tokens $e_1e_2$ with probability $P(e_1e_2)$ 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 $P(f_1|e_1) \cdot P(f_2|e_2)$.

The probability of any cipher bigram $F$ is:

$$P(F) = \sum_{e_1e_2} P(e_1e_2) \prod_{i=1}^{2} P(f_i|e_i)$$

Given a corpus of $N$ cipher bigrams $F_1...F_N$, the probability of the corpus is:

$$P(corpus) = \prod_{j=1}^{N} P(F_j)$$

Given a plaintext bigram language model, the goal is to manipulate $P(f|e)$ 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.

Ravi and Knight (2011) apply Bayesian learning to reduce the space complexity. Instead of estimating probabilities $P(f|e)$, Bayesian learning tries to draw samples from plaintext sequences given ciphertext bigrams. During sampling, the probability of any possible plaintext sample $e_1e_2$ is given as:

$$P_{sample}(e_1e_2) = P(e_1e_2) \prod_{i=1}^{2} P_{bayes}(f_i|e_i)$$

| misión de naciones unidas en oriente medio | |
|---|---|
| misión de | misión naciones |
| de naciones | naciones unidas |
| naciones unidas | misión en |
| unidas en | en oriente |
| en oriente | oriente medio |
| oriente medio | |

Table 1: Comparison of adjacent bigrams (left) and dependency bigrams (right) extracted from the same Spanish text

with $P_{bayes}(f_i|e_i)$ defined as:

$$P_{bayes}(f_i|e_i) = \frac{\alpha P_0(f_i|e_i) + count(f_i, e_i)}{\alpha + count(e_i)}$$

where $P_0$ is a base distribution, and $\alpha$ is a parameter that controls how much we trust $P_0$. $count(f_i, e_i)$ and $count(e_i)$ record the number of times $f_i, e_i$ and $e_i$ appear in previously generated samples respectively.

At the end of sampling, $P(f_i|e_i)$ is estimated by:

$$P(f_i|e_i) = \frac{count(f_i, e_i)}{count(e_i)}$$

However, Bayesian decipherment is still very slow with Gibbs sampling (Geman and Geman, 1987), as each sampling step requires considering $V_e$ possibilities. Dou and Knight (2012) solve the problem by introducing slice sampling (Neal, 2000) to Bayesian decipherment.

## 3 From Adjacent Bigrams to Dependency Bigrams

A major limitation of work by Dou and Knight (2012) is their monotonic generative story for deciphering adjacent bigrams. While the generation process works well for deciphering similar languages (e.g. Spanish and French) without considering reordering, it does not work well for languages that are more different in grammar and word order (e.g. Spanish and English). In this section, we first look at why adjacent bigrams are bad for decipherment. Then we describe how to use syntax to solve the problem.

The left column in Table 1 contains adjacent bigrams extracted from the Spanish phrase "misión de naciones unidas en oriente medio". The correct decipherment for the bigram "naciones unidas" should be "united nations". Since the deciphering model described by Dou and Knight (2012) does not consider word reordering, it needs to decipher the bigram into "nations united" in order to get the right word translations "naciones"→"nations" and "unidas"→"united". However, the English language model used for decipherment is built from English adjacent bigrams, so it strongly disprefers "nations united" and is not likely to produce a sensible decipherment for "naciones unidas". The Spanish bigram "oriente medio" poses the same problem. Thus, without considering word reordering, the model described by Dou and Knight (2012) is not a good fit for deciphering Spanish into English.

However, if we extract bigrams based on dependency relations for both languages, the model fits better. To extract such bigrams, we first use dependency parsers to parse both languages, and extract bigrams by putting head word first, followed by the modifier.[1] We call these dependency bigrams. The right column in Table 1 lists examples of Spanish dependency bigrams extracted from the same Spanish phrase. With a language model built with English dependency bigrams, the same model used for deciphering adjacent bigrams is able to decipher Spanish dependency bigram "naciones(head) unidas(modifier)" into "nations(head) united(modifier)".

We might instead propose to consider word reordering when deciphering adjacent bigrams (e.g. add an operation to swap tokens in a bigram). However, using dependency bigrams has the following advantages:

- First, using dependency bigrams avoids complicating the model, keeping deciphering efficient and scalable.
- Second, it addresses the problem of long distance reordering, which can not be modeled by swapping tokens in bigrams.

Furthermore, using dependency bigrams allows us to use dependency types to further

[1] As use of "del" and "de" in Spanish is much more frequent than the use of "of" in English, we skip those words by using their head words as new heads if any of them serves as a head.

improve decipherment. Suppose we have a Spanish dependency bigram "aceptó(verb) solicitud(object)". Then all of the following English dependency bigrams are possible decipherments: "accepted(verb) UN(subject)", "accepted(verb) government(subject)", "accepted(verb) request(object)". However, if we know the type of the Spanish dependency bigram and use a language model built with the same type in English, the only possible decipherment is "accepted(verb) request(object)". If we limit the search space, a system is more likely to find a better decipherment.

## 4 Deciphering Spanish Gigaword

In this section, we compare dependency bigrams with adjacent bigrams for deciphering Spanish into English.

### 4.1 Data

We use the Gigaword corpus for our decipherment experiments. The corpus contains news articles from different news agencies and is available in Spanish and English. We use only the AFP (Agence France-Presse) section of the corpus in decipherment experiments. We tokenize the corpus using tools that come with the Europarl corpus (Koehn, 2005). To shorten the time required for running different systems on large amounts of data, we keep only the top 5000 most frequent word types in both languages and replace all other word types with UNK. We also throw away lines with more than 40 tokens, as the Spanish parser (Bohnet, 2010) we use is slow when processing long sentences. After preprocessing, the corpus contains approximately 440 million tokens in Spanish and 350 million tokens in English. To obtain dependency bigrams, we use the Bohnet parsers (Bohnet, 2010) to parse both the Spanish and English version of the corpus.

### 4.2 Systems

Three systems are evaluated in the experiments. We implement a baseline system, **Adjacent**, based on Dou and Knight (2012). The baseline system collects adjacent bigrams and their counts from Spanish and English texts. It then builds an English bigram language model using the English adjacent bigrams and uses it to decipher the Spanish adjacent bigrams.

| | Dependency Types |
|---|---|
| Group 1 | Verb/Subject |
| Group 2 | Preposition/Preposition-Object, Noun/Noun-Modifier |
| Group 3 | Verb/Noun-Object |

Table 2: Dependency relations divided into three groups

We build the second system, **Dependency**, using dependency bigrams for decipherment. As the two parsers do not output the same set of dependency relations, we cannot extract all types of dependency bigrams. Instead, we select a subset of dependency bigrams whose dependency relations are shared by the two parser outputs. The selected dependency relations are: Verb/Subject, Verb/Noun-Object, Preposition/Object, Noun/Modifier. Decipherment runs the same way as in the baseline system.

The third system, **DepType**, is built using both dependent bigrams and their dependency types. We first extract dependency bigrams for both languages, then group them based on their dependency types. As both parsers treat noun phrases dependent on "del", "de", and "of" as prepositional phrases, we choose to divide the dependency bigrams into 3 groups and list them in Table 2. A separate language model is built for each group of English dependency bigrams and used to decipher the group of Spanish dependency bigrams with same dependency type.

For all the systems, language models are built using the SRILM toolkit (Stolcke, 2002). For the Adjacent system, we use Good-Turing smoothing. For the other systems, we use a mix of Witten-Bell and Good-Turing smoothing.

### 4.3 Sampling Procedure

In experiments, we find that the iterative sampling method described by Dou and Knight (2012) helps improve deciphering accuracy. We also find that combining results from different decipherments helps find more correct translations at each iteration. Thus, instead of using a single sampling process, we use 10 different sampling processes at each iteration. The details of the new sampling procedure are provided here:

- Extract dependency bigrams from parsing outputs and collect their counts.

- Keep bigrams whose counts are greater than a threshold $\alpha$. Then start 10 different randomly seeded and initialized sampling processes. Perform sampling.
- At the end of sampling, extract word translation pairs $(f, e)$ from the final sample. Estimate translation probabilities $P(e|f)$ for each pair. Then construct a translation table by keeping translation pairs $(f, e)$ seen in more than one decipherment and use the average $P(e|f)$ as the new translation probability.
- Lower the threshold $\alpha$ to include more bigrams into the sampling process. Start 10 different sampling processes again and initialize the first sample using the translation pairs obtained from the previous step (for each Spanish token f, choose an English token e whose $P(e|f)$ is the highest). Perform sampling again.
- Repeat until $\alpha = 1$.

### 4.4 Deciphering Accuracy

We choose the first 1000 lines of the monolingual Spanish texts as our test data. The data contains 37,505 tokens and 6556 word types. We use type accuracy as our evaluation metric: Given a word type $f$ in Spanish, we find a translation pair $(f, e)$ with the highest average $P(e|f)$ from the translation table learned through decipherment. If the translation pair $(f, e)$ can also be found in a gold translation lexicon $T_{gold}$, we treat the word type $f$ as correctly deciphered. Let $|C|$ be the number of word types correctly deciphered, and $|V|$ be the total number of word types evaluated. We define type accuracy as $\frac{|C|}{|V|}$.

To create $T_{gold}$, we use GIZA (Och and Ney, 2003) to align a small amount of Spanish-English parallel text (1 million tokens for each language), and use the lexicon derived from the alignment as our gold translation lexicon. $T_{gold}$ contains a subset of 4408 types seen in the test data, among which, 2878 are also top 5000 frequent word types.

### 4.5 Results

During decipherment, we gradually increase the size of Spanish texts and compare the learning curves of three deciphering systems in Figure 2.

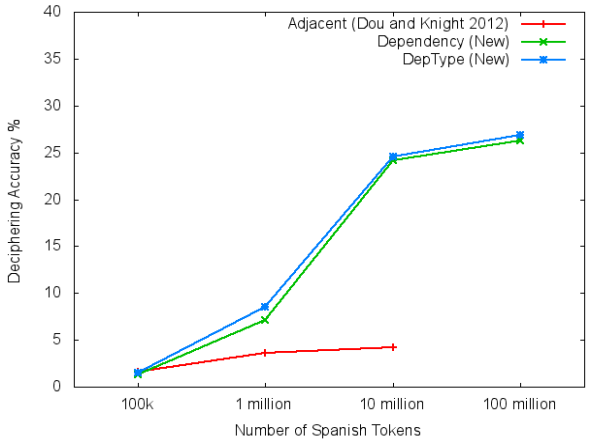


Figure 2: Learning curves for three decipherment systems. Compared with Adjacent (previous state of the art), systems that use dependency bigrams improve deciphering accuracy by over 500%.

With 100k tokens of Spanish text, the performance of the three systems are similar. However, the learning curve of Adjacent plateaus quickly, while those of the dependency based systems soar up as more data becomes available and still rise sharply when the size of Spanish texts increases to 10 million tokens, where the DepType system improves deciphering accuracy of the Adjacent system from 4.2% to 24.6%. In the end, with 100 million tokens, the accuracy of the DepType system rises to 27.0%. The accuracy is even higher (41%), when evaluated against the top 5000 frequent word types only.

## 5 Improving Machine Translation with Decipherment

In this section, we demonstrate how to use a translation lexicon learned by deciphering large amounts of in-domain (news) monolingual data to improve a phrase-based machine translation system trained with limited out-of-domain (politics) parallel data.

### 5.1 Data

We use approximately one million tokens of the Europarl corpus (Koehn, 2005) as our small out-of-domain parallel training data and Gigaword as our large in-domain monolingual training data to build language models and a new translation lexicon to improve a phrase-based MT baseline system. For tuning and testing, we use the development data

| Parallel | | |
|---|---|---|
| | Spanish | English |
| Europarl | 1.1 million | 1.0 million |
| Tune-2008 | 52.6k | 49.8k |
| Test-2009 | 68.1k | 65.6k |
| Test-2010 | 65.5k | 61.9k |
| Test-2011 | 79.4k | 74.7k |
| Non Parallel | | |
| | Spanish | English |
| Gigaword | 894 million | 940 million |

Table 3: Size of training, tuning, and testing data in number of tokens

from the NAACL 2012 workshop on statistical machine translation. The data contains test data in the news domain from the 2008, 2009, 2010, and 2011 workshops. We use the 2008 test data for tuning and the rest for testing. The sizes of the training, tuning, and testing sets are listed in Table 3.

## 5.2 Systems

### 5.2.1 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 build a 5-gram language model using the AFP section of the English Gigaword. We train the other models using the Europarl corpus. By default, Moses uses the following 8 features to score a candidate translation:

- direct and inverse translation probabilities
- direct and inverse lexical weighting
- a language model score
- a distortion score
- phrase penalty
- word penalty

The 8 features have weights adjusted on the tuning data using minimum error rate training (MERT) (Och, 2003). PBMT has a phrase table $T_{phrase}$. During decoding, Moses copies out-of-vocabulary (OOV) words, which can not be found in $T_{phrase}$, directly to output. In the following sections, we describe how to use a translation lexicon learned from large amounts of non parallel data to improve translation of OOV words, as well as words observed in $T_{phrase}$.

### 5.2.2 Decipherment for Machine Translation

To achieve better decipherment, we:

- Increase the size of Spanish ciphertext from 100 million tokens to 894 million tokens.
- Keep top 50k instead of top 5k most frequent word types of the ciphertext.
- Instead of seeding the sampling process randomly, we use a translation lexicon learned from a limited amount of parallel data as seed: For each Spanish dependency bigram $f_1, f_2$, where both $f_1$ and $f_2$ are found in the seed lexicon, we find the English sequence $e_1, e_2$ that maximizes $P(e_1, e_2)P(e_1|f_1)P(e_2|f_2)$. Otherwise, for any Spanish token $f$ that can be found in the seed lexicon, we choose English word $e$, where $P(e|f)$ is the highest as the initial sample; for any $f$ that are not seen in the seed lexicon, we do random initialization.

We perform 20 random restarts with 10k iterations on each and build a word-to-word translation lexicon $T_{decipher}$ by collecting translation pairs seen in at least 3 final decipherments with either $P(f|e) \geq 0.2$ or $P(e|f) \geq 0.2$.

### 5.2.3 Improving Translation of Observed Words with Decipherment

To improve translation of words observed in our parallel corpus, 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 denote this system as **Decipher-OBSV**.

#### 5.2.4 Improving OOV translation with Decipherment

As $T_{decipher}$ is learned from large amounts of in-domain monolingual data, we expect that $T_{decipher}$ contains a number of useful translations for words not seen in the limited amount of parallel data (OOV words). Instead of copying OOV words directly to output, which is what Moses does by default, we try to find translations from $T_{decipher}$ to improve translation.

During decoding, if a source word $f$ is in $T_{phrase}$, its translation options are collected from $T_{phrase}$ exclusively. If $f$ is not in $T_{phrase}$ but in $T_{decipher}$, the decoder will find translations from $T_{decipher}$. If $f$ is not in either translation table, the decoder just copies it directly to the output. We call this system **Decipher-OOV**.

However, when an OOV's correct translation is same as its surface form and all its possible translations in $T_{decipher}$ are wrong, it is better to just copy OOV words directly to output. This scenario happens frequently, as Spanish and English share many common words. To avoid over trusting $T_{decipher}$, we add a new translation pair $(f, f)$ for each source word $f$ in $T_{decipher}$ if the translation pair $(f, f)$ is not originally in $T_{decipher}$. For each newly added translation pair, both of its log translation probabilities are set to 0. To distinguish the added translation pairs from the others learned through decipherment, we add a binary feature $\theta$ to each translation pair in $T_{decipher}$. The final version of $T_{decipher}$ has three feature scores: $P(e|f)$, $P(f|e)$, and $\theta$. Finally, we tune weights of the features in $T_{decipher}$ using MERT (Och, 2003) on the tuning set.

#### 5.2.5 A Combined Approach

In the end, we build a system **Decipher-COMB**, which uses $T_{decipher}$ to improve translation of both observed and OOV words with methods described in sections 5.2.3 and 5.2.4.

### 5.3 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 the translation lexicon learned from decipherment helps achieve higher BLEU scores across tuning and testing sets. **Decipher-OBSV** improves BLEU scores by as much as 1.2 points. We analyze the results and find the gain mainly comes from two parts. First, adding $T_{decipher}$ to small amounts of parallel corpus improves word level translation probabilities, which lead to better lexical weighting; second, $T_{decipher}$ contains new alternative translations for words observed in the parallel corpus.

Moreover, **Decipher-OOV** also achieves better BLEU scores compared with PBMT across all tuning and test sets. We also observe that systems using $T_{decipher}$ learned by deciphering dependency bigrams leads to larger gains in BLEU scores. When decipherment is used to improve translation of both observed and OOV words, we see improvement in BLEU score as high as 1.8 points on the 2010 news test set.

The consistent improvement on the tuning and different testing data suggests that decipherment is capable of learning good translations for a number of OOV words. To further demonstrate that our decipherment approach finds useful translations for OOV words, we list the top 10 most frequent OOV words from both the tuning set and testing set as well as their translations (up to three most likely translations) in Table 5. $P(e|f)$ and $P(f|e)$ are average scores over different decipherment runs.

From the table, we can see that decipherment finds correct translations (bolded) for 7 out of the 10 most frequent OOV words. Moreover, many OOVs and their correct translations are homographs , which makes copying OOVs directly to the output a strong baseline to beat. Nonetheless, decipherment still finds enough correct translations to improve the baseline.

## 6 Conclusion

We introduce syntax for deciphering Spanish into English. Experiment results show that using dependency bigrams improves decipherment accuracy by over 500% compared with the state-of-the-art approach. Moreover, we learn a domain specific translation lexicon by deciphering large amounts of monolingual data and show that the lexicon can improve a baseline machine translation system trained with limited parallel data.

| Decipherment | System | $Tune_{2008}$ | $Test_{2009}$ | $Test_{2010}$ | $Test_{2011}$ |
|---|---|---|---|---|---|
| None | PBMT (Baseline) | 19.1 | 19.6 | 21.3 | 22.1 |
| Adjacent | Decipher-OBSV | 19.5 | 20.1 | 22.2 | 22.6 |
| | Decipher-OOV | 19.4 | 19.9 | 21.7 | 22.5 |
| | Decipher-COMB | 19.5 | 20.2 | 22.3 | 22.5 |
| Dependency | Decipher-OBSV | 19.7 | 20.5 | 22.5 | 23.0 |
| | Decipher-OOV | 19.9 | 20.4 | 22.4 | 22.9 |
| | Decipher-COMB | 20.0 | 20.8 | 23.1 | 23.4 |

Table 4: Systems that use translation lexicons learned from decipherment show consistent improvement over the baseline system across tuning and testing sets. The best system, Decipher-COMB, achieves as much as 1.8 BLEU point gain on the 2010 news test set.

| Spanish | English | $P(e\|f)$ | $P(f\|e)$ |
|---|---|---|---|
| obama | his | 0.33 | 0.01 |
| | bush | 0.27 | 0.07 |
| | clinton | 0.23 | 0.11 |
| bush | **bush** | 0.47 | 0.45 |
| | yeltsin | 0.28 | 0.81 |
| | he | 0.24 | 0.05 |
| festival | event | 0.68 | 0.35 |
| | **festival** | 0.61 | 0.72 |
| wikileaks | zeta | 0.03 | 0.33 |
| venus | **venus** | 0.61 | 0.74 |
| | serena | 0.47 | 0.62 |
| colchones | **mattresses** | 0.55 | 0.73 |
| | cars | 0.31 | 0.01 |
| helado | frigid | 0.52 | 0.44 |
| | chill | 0.37 | 0.14 |
| | sandwich | 0.42 | 0.27 |
| google | microsoft | 0.67 | 0.18 |
| | **google** | 0.59 | 0.69 |
| cantante | **singer** | 0.44 | 0.92 |
| | jackson | 0.14 | 0.33 |
| | artists | 0.14 | 0.77 |
| mccain | **mccain** | 0.66 | 0.92 |
| | it | 0.22 | 0.00 |
| | he | 0.21 | 0.00 |

Table 5: Decipherment finds correct translations for 7 out of 10 most frequent OOV word types.

## 7 Acknowledgments

This work was supported by NSF Grant 0904684 and ARO grant W911NF-10-1-0533. The authors would like to thank David Chiang, Malte Nuhn, Victoria Fossum, Ashish Vaswani, Ulf Hermjakob, Yang Gao, and Hui Zhang (in no particular order) for their comments and suggestions.

## 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 Volume Three*. AAAI Press.

Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics*. Coling.

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.

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.

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.

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.

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.

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.

Philipp Koehn. 2005. Europarl: a parallel corpus for statistical machine translation. In *In Proceedings of the Tenth Machine Translation Summit*, Phuket, Thailand. Asia-Pacific Association for Machine Translation.

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.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*

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

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*.