Bayesian Inference for Zodiac and Other Homophonic Ciphers

Sujith Ravi and Kevin Knight
University of Southern California
Information Sciences Institute
Marina del Rey, California 90292
{sravi,knight}@isi.edu

Abstract

We introduce a novel Bayesian approach for deciphering complex substitution ciphers. Our method uses a decipherment model which combines information from letter n-gram language models as well as word dictionaries. Bayesian inference is performed on our model using an efficient sampling technique. We evaluate the quality of the Bayesian decipherment output on simple and homophonic letter substitution ciphers and show that unlike a previous approach, our method consistently produces almost 100% accurate decipherments. The new method can be applied on more complex substitution ciphers and we demonstrate its utility by cracking the famous Zodiac-408 cipher in a fully automated fashion, which has never been done before.

1 Introduction

Substitution ciphers have been used widely in the past to encrypt secrets behind messages. These ciphers replace (English) plaintext letters with cipher symbols in order to generate the ciphertext sequence.

There exist many published works on automatic decipherment methods for solving simple letter-substitution ciphers. Many existing methods use dictionary-based attacks employing huge word dictionaries to find plaintext patterns within the ciphertext (Peleg and Rosenfeld, 1979; Ganesan and Sherman, 1993; Jakobsen, 1995; Olson, 2007). Most of these methods are heuristic in nature and search for the best deterministic key during decipherment. Others follow a probabilistic decipherment approach. Knight et al. (2006) use the Expectation Maximization (EM) algorithm (Dempster et al., 1977) to search for the best probabilistic key using letter n-gram models. Ravi and Knight (2008) formulate decipherment as an integer programming problem and provide an exact method to solve simple substitution ciphers by using letter n-gram models along with deterministic key constraints. Corlett and Penn (2010) work with large ciphertexts containing thousands of characters and provide another exact decipherment method using an A* search algorithm. Diaconis (2008) presents an analysis of Markov Chain Monte Carlo (MCMC) sampling algorithms and shows an example application for solving simple substitution ciphers.

Most work in this area has focused on solving simple substitution ciphers. But there are variants of substitution ciphers, such as homophonic ciphers, which display increasing levels of difficulty and present significant challenges for decipherment. The famous Zodiac serial killer used one such cipher system for communication. In 1969, the killer sent a three-part cipher message to newspapers claiming credit for recent shootings and crimes committed near the San Francisco area. The 408-character message (Zodiac-408) was manually decoded by hand in the 1960's. Oranchak (2008) presents a method for solving the Zodiac-408 cipher automatically with a dictionary-based attack using a genetic algorithm. However, his method relies on using plaintext words from the known solution to solve the cipher, which departs from a strict decipherment scenario.

In this paper, we introduce a novel method for solving substitution ciphers using Bayesian learning. Our novel contributions are as follows:

- We present a new probabilistic decipherment approach using Bayesian inference with sparse priors, which can be used to solve different types of substitution ciphers.
- Our new method combines information from word dictionaries along with letter n-gram models, providing a robust decipherment model which offsets the disadvantages faced by previous approaches.
- We evaluate the Bayesian decipherment output on three different types of substitution ciphers and show that unlike a previous approach, our new method solves all the ciphers completely.
- Using the Bayesian decipherment, we show for the first time a truly automated system that successfully solves the Zodiac-408 cipher.

2 Letter Substitution Ciphers

We use natural language processing techniques to attack letter substitution ciphers. In a letter substitution cipher, every letter p in the natural language (plaintext) sequence is replaced by a cipher token c, according to some substitution key.

For example, an English plaintext

“H E L L O _ W O R L D ...”

may be enciphered as:

“N O E E I _ T I M E L ...”

according to the key:

p: ABCDEFGHIJKLMNOPQRSTUVWXYZ_
c: XYZLOHANBCDEFGIJKMPQRSTUVW_

where, “_” represents the space character (word boundary) in the English and ciphertext messages.

If the recipients of the ciphertext message have the substitution key, they can use it (in reverse) to recover the original plaintext. The sender can encrypt the message using one of many different cipher systems. The particular type of cipher system chosen determines the properties of the key. For example, the substitution key can be deterministic in both the encipherment and decipherment directions as shown in the above example—i.e., there is a 1-to-1 correspondence between the plaintext letters and ciphertext symbols. Other types of keys exhibit nondeterminism either in the encipherment (or decipherment) or both directions.

2.1 Simple Substitution Ciphers

The key used in a simple substitution cipher is deterministic in both the encipherment and decipherment directions, i.e., there is a 1-to-1 mapping between plaintext letters and ciphertext symbols. The example shown earlier depicts how a simple substitution cipher works.

Data: In our experiments, we work with a 414-letter simple substitution cipher. We encrypt an original English plaintext message using a randomly generated simple substitution key to create the ciphertext. During the encipherment process, we preserve spaces between words and use this information for decipherment—i.e., plaintext character “_” maps to ciphertext character “_”. Figure 1 (top) shows a portion of the ciphertext along with the original plaintext used to create the cipher.

2.2 Homophonic Ciphers

A homophonic cipher uses a substitution key that maps a plaintext letter to more than one cipher symbol.

For example, the English plaintext:

“H E L L O _ W O R L D ...”

may be enciphered as:

“65 82 51 84 05 _ 60 54 42 51 45 ...”

according to the key:

A: 09 12 33 47 53 67 78 92
B: 48 81
...
E: 14 16 24 44 46 55 57 64 74 82 87
...
L: 51 84
...
Z: 02

Here, “_” represents the space character in both English and ciphertext. Notice the non-determinism involved in the enciphering direction—the English letter “L” is substituted using different symbols (51, 84) at different positions in the ciphertext.

These ciphers are more complex than simple substitution ciphers. Homophonic ciphers are generated via a non-deterministic encipherment process—the key is 1-to-many in the enciphering direction. The number of potential cipher symbol substitutes for a particular plaintext letter is often proportional to the frequency of that letter in the plaintext language—for example, the English letter “E” is assigned more cipher symbols than “Z”. The objective of this is to flatten out the frequency distribution of ciphertext symbols, making a frequency-based cryptanalysis attack difficult.

The substitution key is, however, deterministic in the decipherment direction—each ciphertext symbol maps to a single plaintext letter. Since the ciphertext can contain more than 26 types, we need a larger alphabet system—we use a numeric substitution alphabet in our experiments.

Data: For our decipherment experiments on homophonic ciphers, we use the same 414-letter English plaintext used in Section 2.1. We encrypt this message using a homophonic substitution key (available from *http://www.simonsingh.net/The_Black_Chamber/homophoniccipher.htm*). As before, we preserve spaces between words in the ciphertext. Figure 1 (middle) displays a section of the homophonic cipher (with spaces) and the original plaintext message used in our experiments.

2.3 Homophonic Ciphers without spaces (Zodiac-408 cipher)

In the previous two cipher systems, the word-boundary information was preserved in the cipher. We now consider a more difficult homophonic cipher by removing space characters from the original plaintext.

The English plaintext from the previous example now looks like this:

“HELLOWORLD ...”

and the corresponding ciphertext is:

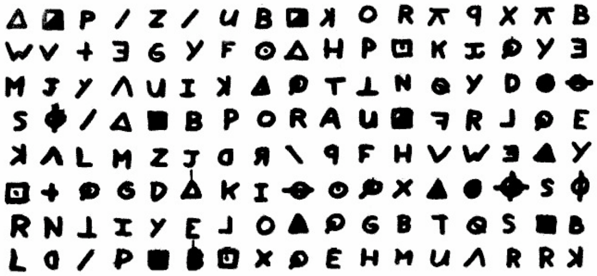
“65 82 51 84 05 60 54 42 51 45 ...”

Without the word boundary information, typical dictionary-based decipherment attacks fail on such ciphers.

Zodiac-408 cipher: Homophonic ciphers without spaces have been used extensively in the past to encrypt secret messages. One of the most famous homophonic ciphers in history was used by the infamous Zodiac serial killer in the 1960’s. The killer sent a series of encrypted messages to newspapers and claimed that solving the ciphers would reveal clues to his identity. The identity of the Zodiac killer remains unknown to date. However, the mystery surrounding this has sparked much interest among cryptanalysis experts and amateur enthusiasts.

The Zodiac messages include two interesting ciphers: (1) a 408-symbol homophonic cipher without spaces (which was solved manually by hand), and (2) a similar looking 340-symbol cipher that has yet to be solved.

Here is a sample of the Zodiac-408 cipher message:



...

and the corresponding section from the original English plaintext message:

```
I L I K E K I L L I N G P E O P L
E B E C A U S E I T I S S O M U C
H F U N I T I S M O R E F U N T H
A N K I L L I N G W I L D G A M E
I N T H E F O R R E S T B E C A U
S E M A N I S T H E M O S T D A N
G E R O U E A N A M A L O F A L L
T O K I L L S O M E T H I N G G I
                ...
```

Besides the difficulty with missing word boundaries and non-determinism associated with the key, the Zodiac-408 cipher poses several additional challenges which makes it harder to solve than any standard homophonic cipher. There are spelling mistakes in the original message (for example, the English word “PARADISE” is misspelt as

"PARADICE") which can divert a dictionary-based attack. Also, the last 18 characters of the plaintext message does not seem to make any sense ("EBEORIETEMETHHPITI").

Data: Figure 1 (bottom) displays the Zodiac-408 cipher (consisting of 408 tokens, 54 symbol types) along with the original plaintext message. We run the new decipherment method (described in Section 3.1) and show that our approach can successfully solve the Zodiac-408 cipher.

3 Decipherment

Given a ciphertext message $c_1...c_n$, the goal of decipherment is to uncover the hidden plaintext message $p_1...p_n$. The size of the keyspace (i.e., number of possible key mappings) that we have to navigate during decipherment is huge—a simple substitution cipher has a keyspace size of 26!, whereas a homophonic cipher such as the Zodiac-408 cipher has 26^{54} possible key mappings.

Next, we describe a new Bayesian decipherment approach for tackling substitution ciphers.

3.1 Bayesian Decipherment

Bayesian inference methods have become popular in natural language processing (Goldwater and Griffiths, 2007; Finkel et al., 2005; Blunsom et al., 2009; Chiang et al., 2010). Snyder et al. (2010) proposed a Bayesian approach in an archaeological decipherment scenario. These methods are attractive for their ability to manage uncertainty about model parameters and allow one to incorporate prior knowledge during inference. A common phenomenon observed while modeling natural language problems is *sparsity*. For simple letter substitution ciphers, the original substitution key exhibits a 1-to-1 correspondence between the plaintext letters and cipher types. It is not easy to model such information using conventional methods like EM. But we can easily specify priors that favor sparse distributions within the Bayesian framework.

Here, we propose a novel approach for deciphering substitution ciphers using Bayesian inference. Rather than enumerating all possible keys (26! for a simple substitution cipher), our Bayesian framework requires us to sample only a small number of keys during the decipherment process.

Probabilistic Decipherment: Our decipherment method follows a noisy-channel approach. We are faced with a ciphertext sequence $c = c_1...c_n$ and we want to find the (English) letter sequence $p = p_1...p_n$ that maximizes the probability $P(p|c)$.

We first formulate a generative story to model the process by which the ciphertext sequence is generated.

1. Generate an English plaintext sequence $p = p_1...p_n$, with probability $P(p)$.

2. Substitute each plaintext letter p_i with a ciphertext token c_i, with probability $P(c_i|p_i)$ in order to generate the ciphertext sequence $c = c_1...c_n$.

We build a statistical English language model (LM) for the plaintext source model $P(p)$, which assigns a probability to any English letter sequence. Our goal is to estimate the channel model parameters θ in order to maximize the probability of the observed ciphertext c:

$$\arg\max_{\theta} P(c) = \arg\max_{\theta} \sum_{p} P_{\theta}(p, c) \qquad (1)$$

$$= \arg\max_{\theta} \sum_{p} P(p) \cdot P_{\theta}(c|p) \qquad (2)$$

$$= \arg\max_{\theta} \sum_{p} P(p) \cdot \prod_{i=1}^{n} P_{\theta}(c_i|p_i) \qquad (3)$$

We estimate the parameters θ using Bayesian learning. In our decipherment framework, a Chinese Restaurant Process formulation is used to model both the source and channel. The detailed generative story using CRPs is shown below:

1. $i \leftarrow 1$

2. Generate the English plaintext letter p_1, with probability $P_0(p_1)$

3. Substitute p_1 with cipher token c_1, with probability $P_0(c_1|p_1)$

4. $i \leftarrow i + 1$

5. Generate English plaintext letter p_i, with probability

$$\frac{\alpha \cdot P_0(p_i|p_{i-1}) + C_1^{i-1}(p_{i-1}, p_i)}{\alpha + C_1^{i-1}(p_{i-1})}$$

Plaintext:	D E C I P H E R M E N T _ I S _ T H E _ A N A L Y S I S _ O F _ D O C U M E N T S _ W R I T T E N _ I N _ A N C I E N T _ L A N G U A G E S _ W H E R E _ T H E _ ...
Ciphertext:	i n g c m p n q s n w f _ c v _ f p n _ o w o k t v c v _ h u _ i h g z s n w f v _ r q c f f n w _ c w _ o w g c n w f _ k o w a z o a n v _ r p n q n _ f p n _ ...
Bayesian solution:	D E C I P H E R M E N T _ I S _ T H E _ A N A L Y S I S _ O F _ D O C U M E N T S _ W R I T T E N _ I N _ A N C I E N T _ L A N G U A G E S _ W H E R E _ T H E _ ...

Plaintext:	D E C I P H E R M E N T _ I S _ T H E _ A N A L Y S I S _ O F _ D O C U M E N T S _ W R I T T E N _ I N _ ...
Ciphertext:	79 57 62 93 95 68 44 77 22 74 59 97 _ 32 86 _ 85 56 82 _ 67 59 67 84 52 86 73 11 _ 99 10 _ 45 90 13 61 27 98 71 49 19 _ 60 80 88 85 20 55 59 _ 32 91 _ ...
Bayesian solution:	D E C I P H E R M E N T _ I S _ T H E _ A N A L Y S I S _ O F _ D O C U M E N T S _ W R I T T E N _ I N _ ...

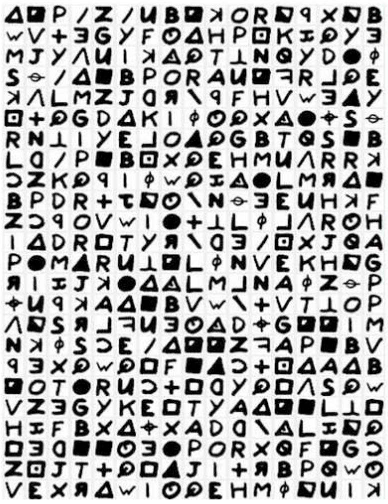
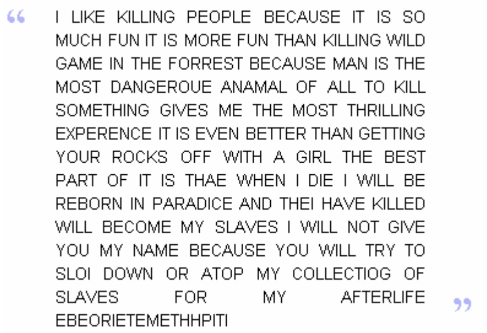
Ciphertext:	
Plaintext:	 “ I LIKE KILLING PEOPLE BECAUSE IT IS SO MUCH FUN IT IS MORE FUN THAN KILLING WILD GAME IN THE FORREST BECAUSE MAN IS THE MOST DANGEROUE ANAMAL OF ALL TO KILL SOMETHING GIVES ME THE MOST THRILLING EXPERENCE IT IS EVEN BETTER THAN GETTING YOUR ROCKS OFF WITH A GIRL THE BEST PART OF IT IS THAE WHEN I DIE I WILL BE REBORN IN PARADICE AND THEI HAVE KILLED WILL BECOME MY SLAVES I WILL NOT GIVE YOU MY NAME BECAUSE YOU WILL TRY TO SLOI DOWN OR ATOP MY COLLECTIOG OF SLAVES FOR MY AFTERLIFE EBEORIETEMETHHPITI ”
Bayesian solution (final decoding):	I L I K E K I L L I N G P E O P L E B E C A U S E I T I S S O M U C H F U N I T I **A** M O R E F U N T H A N K I L L I N G W I L D G A M E I N T H E F O R R E S T B E C A U S E M A N I S T H E M O **A** T D A N G E R **T** U E A N A M A L O F A L L ...
(with spaces shown):	I _ L I K E _ K I L L I N G _ P E O P L E _ B E C A U S E _ I T _ I S _ S O _ M U C H _ F U N _ I T _ I _ **A** _ M O R E _ F U N _ T H A N _ K I L L I N G _ W I L D _ G A M E _ I N _ T H E _ F O R _ R E S T _ B E C A U S E _ M A N _ I S _ T H E _ M O **A** T _ D A N G E R _ **T** U E _ A N _ A M _ A L _ O F _ A L L _ ...

Figure 1: Samples from the ciphertext sequence, corresponding English plaintext message and output from Bayesian decipherment (using word+3-gram LM) for three different ciphers: (a) Simple Substitution Cipher (top), (b) Homophonic Substitution Cipher with spaces (middle), and (c) Zodiac-408 Cipher (bottom).

6. Substitute p_i with cipher token c_i, with probability

$$\frac{\beta \cdot P_0(c_i|p_i) + C_1^{i-1}(p_i, c_i)}{\beta + C_1^{i-1}(p_i)}$$

7. With probability P_{quit}, quit; else go to Step 4.

This defines the probability of any given derivation, i.e., any plaintext hypothesis corresponding to the given ciphertext sequence. The base distribution P_0 represents prior knowledge about the model parameter distributions. For the plaintext source model, we use probabilities from an English language model and for the channel model, we specify a uniform distribution (i.e., a plaintext letter can be substituted with any given cipher type with equal probability). C_1^{i-1} represents the count of events occurring before plaintext letter p_i in the derivation (we call this the "cache"). α and β represent Dirichlet prior hyperparameters over the source and channel models respectively. A large prior value implies that characters are generated from the base distribution P_0, whereas a smaller value biases characters to be generated with reference to previous decisions inside the cache (favoring sparser distributions).

Efficient inference via type sampling: We use a Gibbs sampling (Geman and Geman, 1984) method for performing inference on our model. We could follow a point-wise sampling strategy, where we sample plaintext letter choices for every cipher token, one at a time. But we already know that the substitution ciphers described here exhibit determinism in the deciphering direction,[1] i.e., although we have no idea about the key mappings themselves, we do know that there exists only a single plaintext letter mapping for every cipher symbol type in the true key. So sampling plaintext choices for every cipher token separately is not an efficient strategy—our sampler may spend too much time exploring invalid keys (which map the same cipher symbol to different plaintext letters).

Instead, we use a *type sampling* technique similar to the one proposed by Liang et al. (2010). Under this scheme, we sample plaintext letter choices for each cipher symbol type. In every step, we sample a new plaintext letter for a cipher type and update the entire plaintext hypothesis (i.e., plaintext letters at all corresponding positions) to reflect this change. For example, if we sample a new choice p_{new} for a cipher symbol which occurs at positions $4, 10, 18$, then we update plaintext letters p_4, p_{10} and p_{18} with the new choice p_{new}.

Using the property of exchangeability, we derive an incremental formula for re-scoring the probability of a new derivation based on the probability of the old derivation—when sampling at position i, we pretend that the area affected (within a context window around i) in the current plaintext hypothesis occurs at the end of the corpus, so that both the old and new derivations share the same cache.[2] While we may make corpus-wide changes to a derivation in every sampling step, exchangeability allows us to perform scoring in an efficient manner.

Combining letter n-gram language models with word dictionaries: Many existing probabilistic approaches use statistical letter n-gram language models of English to assign $P(p)$ probabilities to plaintext hypotheses during decipherment. Other decryption techniques rely on word dictionaries (using words from an English dictionary) for attacking substitution ciphers.

Unlike previous approaches, our decipherment method combines information from both sources—letter n-grams and word dictionaries. We build an interpolated *word+n-gram* LM and use it to assign $P(p)$ probabilities to any plaintext letter sequence $p_1 ... p_n$.[3] The advantage is that it helps direct the sampler towards plaintext hypotheses that resemble natural language—high probability letter sequences which form valid words such as "`H E L L O`" instead of sequences like "'`T X H R T`". But in addition to this, using letter n-gram information makes

[1] This assumption does not strictly apply to the Zodiac-408 cipher where a few cipher symbols exhibit non-determinism in the decipherment direction as well.

[2] The relevant context window that is affected when sampling at position i is determined by the word boundaries to the left and right of i.

[3] We set the interpolation weights for the word and n-gram LM as (0.9, 0.1). The word-based LM is constructed from a dictionary consisting of 9,881 frequently occurring words collected from Wikipedia articles. We train the letter n-gram LM on 50 million words of English text available from the Linguistic Data Consortium.

our model robust against variations in the original plaintext (for example, unseen words or misspellings as in the case of Zodiac-408 cipher) which can easily throw off dictionary-based attacks. Also, it is hard for a point-wise (or type) sampler to "find words" starting from a random initial sample, but easier to "find n-grams".

Sampling for ciphers without spaces: For ciphers without spaces, dictionaries are hard to use because we do not know where words start and end. We introduce a new sampling operator which counters this problem and allows us to perform inference using the same decipherment model described earlier. In a first sampling pass, we sample from 26 plaintext letter choices (e.g., "A", "B", "C", ...) for every cipher symbol type as before. We then run a second pass using a new sampling operator that iterates over adjacent plaintext letter pairs p_{i-1}, p_i in the current hypothesis and samples from two choices—(1) add a word boundary (space character "_") between p_{i-1} and p_i, or (2) remove an existing space character between p_{i-1} and p_i.

For example, given the English plaintext hypothesis "`... A B O Y ...`", there are two sampling choices for the letter pair `A,B` in the second step. If we decide to add a word boundary, our new plaintext hypothesis becomes "`... A _ B O Y ...`".

We compute the derivation probability of the new sample using the same efficient scoring procedure described earlier. The new strategy allows us to apply Bayesian decipherment even to ciphers without spaces. As a result, we now have a new decipherment method that consistently works for a range of different types of substitution ciphers.

Decoding the ciphertext: After the sampling run has finished,[4] we choose the final sample as our English plaintext decipherment output.

4 Experiments and Results

We run decipherment experiments on different types of letter substitution ciphers (described in Section 2). In particular, we work with the following three ciphers:

(a) 414-letter Simple Substitution Cipher

(b) 414-letter Homophonic Cipher (with spaces)

(c) Zodiac-408 Cipher

Methods: For each cipher, we run and compare the output from two different decipherment approaches:

1. **EM Method** using letter n-gram LMs following the approach of Knight et al. (2006). They use the EM algorithm to estimate the channel parameters θ during decipherment training. The given ciphertext c is then decoded by using the Viterbi algorithm to choose the plaintext decoding p that maximizes $P(p) \cdot P_\theta(c|p)^3$, stretching the channel probabilities.

2. **Bayesian Decipherment** method using word+n-gram LMs (novel approach described in Section 3.1).

Evaluation: We evaluate the quality of a particular decipherment as the percentage of cipher tokens that are decoded correctly.

Results: Figure 2 compares the decipherment performance for the EM method with Bayesian decipherment (using type sampling and sparse priors) on three different types of substitution ciphers. Results show that our new approach (Bayesian) outperforms the EM method on all three ciphers, solving them completely. Even with a 3-gram letter LM, our method yields a +63% improvement in decipherment accuracy over EM on the homophonic cipher with spaces. We observe that the word+3-gram LM proves highly effective when tackling more complex ciphers and cracks the Zodiac-408 cipher. Figure 1 shows samples from the Bayesian decipherment output for all three ciphers. For ciphers without spaces, our method automatically guesses the word boundaries for the plaintext hypothesis.

[4] For letter substitution decipherment we want to keep the language model probabilities fixed during training, and hence we set the prior on that model to be high ($\alpha = 10^4$). We use a sparse prior for the channel ($\beta = 0.01$). We instantiate a key which matches frequently occurring plaintext letters to frequent cipher symbols and use this to generate an initial sample for the given ciphertext and run the sampler for 5000 iterations. We use a linear annealing schedule during sampling decreasing the temperature from $10 \rightarrow 1$.

Method	**LM**	**Accuracy (%) on 414-letter Simple Substitution Cipher**	**Accuracy (%) on 414-letter Homophonic Substitution Cipher (with spaces)**	**Accuracy (%) on Zodiac-408 Cipher**
1. EM	2-gram	83.6	30.9	–
	3-gram	99.3	32.6	0.3* (*28.8 *with 100 restarts*)
2. Bayesian	3-gram	**100.0**	95.2	23.0
	word+2-gram	**100.0**	**100.0**	–
	word+3-gram	**100.0**	**100.0**	**97.8**

Figure 2: Comparison of decipherment accuracies for EM *versus* Bayesian method when using different language models of English on the three substitution ciphers: (a) 414-letter Simple Substitution Cipher, (b) 414-letter Homophonic Substitution Cipher (with spaces), and (c) the famous Zodiac-408 Cipher.

For the Zodiac-408 cipher, we compare the performance achieved by Bayesian decipherment under different settings:

- *Letter n-gram* versus *Word+n-gram* LMs—Figure 2 shows that using a word+3-gram LM instead of a 3-gram LM results in +75% improvement in decipherment accuracy.
- *Sparse* versus *Non-sparse* priors—We find that using a sparse prior for the channel model (β = 0.01 versus 1.0) helps for such problems and produces better decipherment results (97.8% versus 24.0% accuracy).
- *Type* versus *Point-wise* sampling—Unlike point-wise sampling, type sampling quickly converges to better decipherment solutions. After 5000 sampling passes over the entire data, decipherment output from type sampling scores 97.8% accuracy compared to 14.5% for the point-wise sampling run.[5]

We also perform experiments on shorter substitution ciphers. On a 98-letter simple substitution cipher, EM using 3-gram LM achieves 41% accuracy, whereas the method from Ravi and Knight (2009) scores 84% accuracy. Our Bayesian method performs the best in this case, achieving 100% with word+3-gram LM.

5 Conclusion

In this work, we presented a novel Bayesian decipherment approach that can effectively solve a variety of substitution ciphers. Unlike previous approaches, our method combines information from letter n-gram language models and word dictionaries and provides a robust decipherment model. We empirically evaluated the method on different substitution ciphers and achieve perfect decipherments on all of them. Using Bayesian decipherment, we can successfully solve the Zodiac-408 cipher—the first time this is achieved by a fully automatic method in a strict decipherment scenario.

For future work, there are other interesting decipherment tasks where our method can be applied. One challenge is to crack the unsolved Zodiac-340 cipher, which presents a much harder problem than the solved version.

Acknowledgements

The authors would like to thank the reviewers for their comments. This research was supported by NSF grant IIS-0904684.

References

Phil Blunsom, Trevor Cohn, Chris Dyer, and Miles Osborne. 2009. A Gibbs sampler for phrasal synchronous grammar induction. 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 Asian Federation of Natural Language Processing (ACL-IJCNLP)*, pages 782–790.

David Chiang, Jonathan Graehl, Kevin Knight, Adam Pauls, and Sujith Ravi. 2010. Bayesian inference for finite-state transducers. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL/HLT)*, pages 447–455.

[5] Both sampling runs were seeded with the same random initial sample.

Eric Corlett and Gerald Penn. 2010. An exact A* method for deciphering letter-substitution ciphers. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1040–1047.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.

Persi Diaconis. 2008. The Markov Chain Monte Carlo revolution. *Bulletin of the American Mathematical Society*, 46(2):179–205.

Jenny Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 363–370.

Ravi Ganesan and Alan T. Sherman. 1993. Statistical techniques for language recognition: An introduction and guide for cryptanalysts. *Cryptologia*, 17(4):321–366.

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

Sharon Goldwater and Thomas 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.

Thomas Jakobsen. 1995. A fast method for cryptanalysis of substitution ciphers. *Cryptologia*, 19(3):265–274.

Kevin Knight, Anish Nair, Nishit Rathod, and Kenji Yamada. 2006. Unsupervised analysis for decipherment problems. In *Proceedings of the Joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics*, pages 499–506.

Percy Liang, Michael I. Jordan, and Dan Klein. 2010. Type-based MCMC. In *Proceedings of the Conference on Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 573–581.

Edwin Olson. 2007. Robust dictionary attack of short simple substitution ciphers. *Cryptologia*, 31(4):332–342.

David Oranchak. 2008. Evolutionary algorithm for decryption of monoalphabetic homophonic substitution ciphers encoded as constraint satisfaction problems. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, pages 1717–1718.

Shmuel Peleg and Azriel Rosenfeld. 1979. Breaking substitution ciphers using a relaxation algorithm. *Comm. ACM*, 22(11):598–605.

Sujith Ravi and Kevin Knight. 2008. Attacking decipherment problems optimally with low-order n-gram models. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP)*, pages 812–819.

Sujith Ravi and Kevin Knight. 2009. Probabilistic methods for a Japanese syllable cipher. In *Proceedings of the International Conference on the Computer Processing of Oriental Languages (ICCPOL)*, pages 270–281.

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.