EM Decipherment for Large Vocabularies

Malte Nuhn and **Hermann Ney**
Human Language Technology and Pattern Recognition
Computer Science Department, RWTH Aachen University, Aachen, Germany
<surname>@cs.rwth-aachen.de

Abstract

This paper addresses the problem of EM-based decipherment for large vocabularies. Here, decipherment is essentially a tagging problem: Every cipher token is tagged with some plaintext type. As with other tagging problems, this one can be treated as a Hidden Markov Model (HMM), only here, the vocabularies are large, so the usual $\mathcal{O}(NV^2)$ exact EM approach is infeasible. When faced with this situation, many people turn to sampling. However, we propose to use a type of approximate EM and show that it works well. The basic idea is to collect fractional counts only over a small subset of links in the forward-backward lattice. The subset is different for each iteration of EM. One option is to use beam search to do the subsetting. The second method restricts the successor words that are looked at, for each hypothesis. It does this by consulting pre-computed tables of likely n-grams and likely substitutions.

1 Introduction

The decipherment of probabilistic substitution ciphers (ciphers in which each plaintext token can be substituted by any cipher token, following a distribution $p(f|e)$, cf. Table 2) can be seen as an important step towards decipherment for MT. This problem has not been studied explicitly before. Scaling to larger vocabularies for probabilistic substitution ciphers decipherment is a difficult problem: The algorithms for 1:1 or homophonic substitution ciphers are not applicable, and standard algorithms like EM training become intractable when vocabulary sizes go beyond a few hundred words. In this paper we present an efficient EM based training procedure for probabilistic substitution ciphers which provides high decipherment accuracies while having low computational requirements. The proposed approach allows using high order n-gram language models, and is scalable to large vocabulary sizes. We show improvements in decipherment accuracy in a variety of experiments (including MT) while being computationally more efficient than previous published work on EM-based decipherment.

2 Related Work

Several methods exist for deciphering 1:1 substitution ciphers: Ravi and Knight (2008) solve 1:1 substitution ciphers by formulating the decipherment problem as an integer linear program. Corlett and Penn (2010) solve the same problem using A^* search. Nuhn et al. (2013) present a beam search approach that scales to large vocabulary and high order language models. Even though being successful, these algorithms are not applicable to probabilistic substitution ciphers, or any of its extensions as they occur in decipherment for machine translation.

EM training for probabilistic ciphers was first covered in Ravi and Knight (2011). Nuhn et al. (2012) have given an approximation to exact EM training using context vectors, allowing to training models even for larger vocabulary sizes. Ravi (2013) report results on the OPUS subtitle corpus using an elaborate hash sampling technique, based on n-gram language models and context vectors, that is computationally very efficient.

Conventional beam search is a well studied topic: Huang et al. (1992) present beam search for automatic speech recognition, using fine-grained pruning procedures. Similarly, Young and Young (1994) present an HMM toolkit, including pruned forward-backward EM training. Pal et al. (2006) use beam search for training of CRFs.

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 759–764,
Baltimore, Maryland, USA, June 23-25 2014.

Method	Publications	Complexity
EM Full	(Knight et al., 2006), (Ravi and Knight, 2011)	$\mathcal{O}(NV^n)$
EM Fixed Candidates	(Nuhn et al., 2012)	$\mathcal{O}(N)$
EM Beam	This Work	$\mathcal{O}(NV)$
EM Lookahead	This Work	$\mathcal{O}(N)$

Table 1: Different approximations to exact EM training for decipherment. N is the cipher sequence length, V the size of the target vocabulary, and n the order of the language model.

The main contribution of this work is the preselection beam search that—to the best of our knowledge—was not known in literature before, and serves as an important step to applying EM training to the large vocabulary decipherment problem. Table 1 gives an overview of the EM based methods. More details are given in Section 3.2.

3 Probabilistic Substitution Ciphers

We define probabilistic substitutions ciphers using the following generative story for ciphertext sequences f_1^N:

1. Stochastically generate a plaintext sequence e_1^N according to a bigram[1] language model.
2. For each plaintext token e_n choose a substitution f_n with probability $P(f_n|e_n, \vartheta)$.

This generative story corresponds to the model

$$p(e_1^N, f_1^N, \vartheta) = p(e_1^N) \cdot p(f_1^N|e_1^N, \vartheta) \; , \qquad (1)$$

with the zero-order membership model

$$p(f_1^N|e_1^N, \vartheta) = \prod_{n=1}^{N} p_{lex}(f_n|e_n, \vartheta) \qquad (2)$$

with parameters $p(f|e, \vartheta) \equiv \vartheta_{f|e}$ and normalization constraints $\forall e \sum_f \vartheta_{f|e} = 1$, and first-order plaintext sequence model

$$P(e_1^N) = \prod_{n=1}^{N} p_{LM}(e_n|e_{n-1}) \; . \qquad (3)$$

Thus, the probabilistic substitution cipher can be seen as a Hidden Markov Model. Table 2 gives an overview over the model. We want to find those parameters ϑ that maximize the marginal distribution $p(f_1^N|\vartheta)$:

$$\vartheta \; = \; \arg\max_{\vartheta'} \left\{ \sum_{[e_1^N]} p(f_1^N, e_1^N|\vartheta') \right\} \qquad (4)$$

After we obtained the parameters ϑ we can obtain e_1^N as the Viterbi decoding $\arg\max_{e_1^N} \{p(e_1^N|f_1^N, \vartheta)\}$.

3.1 Exact EM training

In the decipherment setting, we are given the observed ciphertext f_1^N and the model $p(f_1^N|e_1^N, \vartheta)$ that explains how the observed ciphertext has been generated given a latent plaintext e_1^N. Marginalizing the unknown e_1^N, we would like to obtain the maximum likelihood estimate of ϑ as specified in Equation 4. We iteratively compute the maximum likelihood estimate by applying the EM algorithm (Dempster et al., 1977):

$$\tilde{\vartheta}_{f|e} = \frac{\sum\limits_{n:f_n=f} p_n(e|f_1^N, \vartheta)}{\sum\limits_{f} \sum\limits_{n:f_n=f} p_n(e|f_1^N, \vartheta)} \qquad (5)$$

with

$$p_n(e|f_1^N, \vartheta) = \sum_{[e_1^N : e_n = e]} p(e_1^N|f_1^N, \vartheta) \qquad (6)$$

being the posterior probability of observing the plaintext symbol e at position n given the ciphertext sequence f_1^N and the current parameters ϑ. $p_n(e|f_1^N, \vartheta)$ can be efficiently computed using the forward-backward algorithm.

3.2 Approximations to EM-Training

The computational complexity of EM training stems from the sum $\sum_{[e_1^N : e_n = e]}$ contained in the posterior $p_n(e|f_1^N, \vartheta)$. However, we can approximate this sum (and hope that the EM training procedure is still working) by only evaluating the dominating terms, i.e. we only evaluate the sum for sequences e_1^N that have the *largest* contributions to $\sum_{[e_1^N : e_n = e]}$. Note that due to this approximation, the new parameter estimates in Equation 5 can become zero. This is a critical issue, since pairs (e, f) with $p(f|e) = 0$ cannot recover from

[1] This can be generalized to n-gram language models.

Sequence of cipher tokens	:	f_1^N	=	$f_1, \ldots, f_N$
Sequence of plaintext tokens	:	e_1^N	=	$e_1, \ldots, e_N$
Joint probability	:	$p(f_1^N, e_1^N \mid \vartheta)$	=	$p(e_1^N) \cdot p(f_1^N \mid e_1^N, \vartheta)$
Language model	:	$p(e_1^N)$	=	$\prod_{n=1}^{N} p_{LM}(e_n \mid e_{n-1})$
Membership probabilities	:	$p(f_1^N \mid e_1^N, \vartheta)$	=	$\prod_{n=1}^{N} p_{lex}(f_n \mid e_n, \vartheta)$
		Paramater Set	:	$\vartheta = \{\vartheta_{f \mid e}\}, p(f \mid e, \vartheta) = \vartheta_{f \mid e}$
		Normalization	:	$\forall e : \sum_{f} \vartheta_{f \mid e} = 1$
Probability of cipher sequence	:	$p(f_1^N \mid \vartheta)$	=	$\sum_{[e_1^N]} p(f_1^N, e_1^N \mid \vartheta)$

Table 2: Definition of the probabilistic substitution cipher model. In contrast to simple or homophonic substitution ciphers, each plaintext token can be substituted by multiple cipher text tokens. The parameter $\vartheta_{f|e}$ represents the probability of substituting token e with token f.

acquiring zero probability in some early iteration. In order to allow the lexicon to recover from these zeros, we use a smoothed lexicon $\hat{p_{lex}}(f|e) = \lambda p_{lex}(f|e) + (1-\lambda)/|V_f|$ with $\lambda = 0.9$ when conducting the E-Step.

3.2.1 Beam Search

Instead of evaluating the sum for terms with the *exact* largest contributions, we restrict ourselves to terms that are *likely* to have a large contribution to the sum, dropping any guarantees about the actual contribution of these terms.

Beam search is a well known algorithm related to this idea: We build up sequences e_1^c with growing cardinality c. For each cardinality, only a set of the B most promising hypotheses is kept. Then for each active hypothesis of cardinality c, all possible extensions with substitutions $f_{c+1} \rightarrow e_{c+1}$ are explored. Then in turn only the best B out of the resulting $B \cdot V_e$ many hypotheses are kept and the algorithm continues with the next cardinality. Reaching the full cardinality N, the algorithm explored $B \cdot N \cdot V_e$ many hypotheses, resulting in a complexity of $\mathcal{O}(BNV_e)$.

Even though EM training using beam search works well, it still suffers from exploring *all* V_e possible extensions for each active hypothesis, and thus scaling linearly with the vocabulary size. Due to that, standard beam search EM training is too slow to be used in the decipherment setting.

3.2.2 Preselection Search

Instead of evaluating all substitutions $f_{c+1} \rightarrow e_{c+1} \in V_e$, this algorithm only expands a fixed number of candidates: For a hypothesis ending in a language model state σ, we only look at B_{LM} many successor words e_{c+1} with the highest LM probability $p_{LM}(e_{c+1}|\sigma)$ and at B_{lex} many successor words e_{c+1} with the highest lexical probability $p_{lex}(f_{c+1}|e_{c+1})$. Altogether, for each hypothesis we only look at $(B_{LM} + B_{lex})$ many successor states. Then, just like in the standard beam search approach, we prune all explored new hypotheses and continue with the pruned set of B many hypotheses. Thus, for a cipher of length N we only explore $N \cdot B \cdot (B_{LM} + B_{lex})$ many hypotheses.[2]

Intuitively speaking, our approach solves the EM training problem for decipherment using large vocabularies by focusing only on those substitutions that either seem likely due to the language model (”What word is likely to follow the current partial decipherment?”) or due to the lexicon model (”Based on my knowledge about the current cipher token, what is the most likely substitution?”).

In order to efficiently find the maximizing e for $p_{LM}(e|\sigma)$ and $p_{lex}(f|e)$, we build a lookup table that contains for each language model state σ the B_{LM} best successor words e, and a separate lookup table that contains for each source word f the B_{lex} highest scoring tokens e. The language model lookup table remains constant during all iterations, while the lexicon lookup table needs to be updated between each iteration.

Note that the size of the LM lookup table scales linearly with the number of language model states. Thus the memory requirements for the lookup ta-

[2] We always use $B = 100$, $B_{lex} = 5$, and $B_{LM} = 50$.

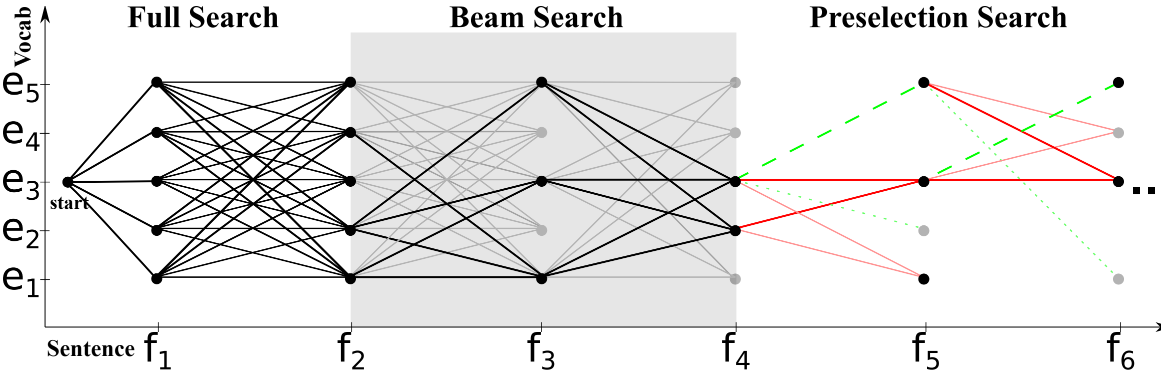

Figure 1: Illustration of the search space explored by full search, beam search, and preselection search. Full search keeps all possible hypotheses at cardinality c and explores all possible substitutions at $(c+1)$. Beam search only keeps the B most promising hypotheses and then selects the best new hypotheses for cardinality $(c+1)$ from *all* possible substitutions. Preselection search keeps only the B best hypotheses for every cardinality c and only looks at the $(B_{lex} + B_{LM})$ most promising substitutions for cardinality $(c+1)$ based on the current lexicon (B_{lex} dashed lines) and language model (B_{LM} solid lines).

Name	Lang.	Sent.	Words	Voc.
VERBMOBIL	English	27,862	294,902	3,723
OPUS	Spanish	13,181	39,185	562
	English	19,770	61,835	411

Table 3: Statistics of the copora used in this paper: The VERBMOBIL corpus is used to conduct experiments on simple substitution ciphers, while the OPUS corpus is used in our Machine Translation experiments.

Vocab	LM	Method	Acc.[%]	Time[h]
200	2	exact	97.19	224.88
		beam	98.87	9.04
		presel.	98.50	4.14
500	2	beam	92.12	24.27
		presel.	92.16	4.70
3 661	3	beam	91.16	302.81
		presel.	90.92	19.68
	4	presel.	92.14	23.72

Table 4: Results for simple substitution ciphers based on the VERBMOBIL corpus using exact, beam, and preselection EM. Exact EM is not tractable for vocabulary sizes above 200.

ble do not form a practical problem of our approach. Figure 1 illustrates full search, beam search, and our proposed method.

4 Experimental Evaluation

We first show experiments for data in which the underlying model is an actual 1:1 substitution cipher. In this case, we report the word accuracy of the final decipherment. We then show experiments for a simple machine translation task. Here we report translation quality in BLEU. The corpora used in this paper are shown in Table 3.

4.1 Simple Substitution Ciphers

In this set of experiments, we compare the exact EM training to the approximations presented in this paper. We use the English side of the German-English VERBMOBIL corpus (Wahlster, 2000) to construct a word substitution cipher, by substituting every word type with a unique number. In order to have a non-parallel setup, we train language models of order 2, 3 and 4 on the first half of the corpus and use the second half as ciphertext. Table 4 shows the results of our experiments.

Since exact EM is not tractable for vocabulary sizes beyond 200 words, we train word classes on the whole corpus and map the words to classes (consistent along the first and second half of the corpus). By doing this, we create new simple substitution ciphers with smaller vocabularies of size 200 and 500. For the smallest setup, we can directly compare all three EM variants. We also include experiments on the original corpus with vocabulary size of 3661. When comparing exact EM training with beam- and preselection EM training, the first thing we notice is that it takes about 20 times longer to run the exact EM training than training with beam EM, and about 50 times longer than the preselection EM training. Interestingly,

Model	Method	BLEU [%]	Runtime
2-gram	Exact EM(Ravi and Knight, 2011)	15.3	850.0h
whole segment lm	Exact EM(Ravi and Knight, 2011)	19.3	850.0h
2-gram	Preselection EM (This work)	15.7	1.8h
3-gram	Preselection EM (This work)	19.5	1.9h

Table 5: Comparison of MT performance (BLEU scores) and efficiency (running time in CPU hours) on the Spanish/English OPUS corpus using only non-parallel corpora for training.

the accuracy of the approximations to exact EM training is better than that of the exact EM training. Even though this needs further investigation, it is clear that the pruned versions of EM training find sparser distributions $p_{lex}(f|e)$: This is desirable in this set of experiments, and could be the reason for improved performance.

For larger vocabularies, exact EM training is not tractable anymore. We thus constrain ourselves to running experiments with beam and preselection EM training only. Here we can see that the runtime of the preselection search is roughly the same as when running on a smaller vocabulary, while the beam search runtime scales almost linearly with the vocabulary size. For the full vocabulary of 3661 words, preselection EM using a 4-gram LM needs less than 7% of the time of beam EM with a 3-gram LM and performs by 1% better in symbol accuracy.

To summarize: Beam search EM is an order of magnitude faster than exact EM training while even increasing decipherment accuracy. Our new preselection search method is in turn orders of magnitudes faster than beam search EM while even being able to outperform exact EM and beam EM by using higher order language models. We were thus able to scale the EM decipherment to larger vocabularies of several thousand words. The runtime behavior is also consistent with the computational complexity discussed in Section 3.2.

4.2 Machine Translation

We show that our algorithm is directly applicable to the decipherment problem for machine translation. We use the same simplified translation model as presented by Ravi and Knight (2011). Because this translation model allows insertions and deletions, hypotheses of different cardinalities coexist during search. We extend our search approach such that pruning is done for each cardinality separately. Other than that, we use the same preselection search procedure as used for the simple substitution cipher task.

We run experiments on the opus corpus as presented in (Tiedemann, 2009). Table 5 shows previously published results using EM together with the results of our new method:

(Ravi and Knight, 2011) is the only publication that reports results using exact EM training and *only* n-gram language models on the target side: It has an estimated runtime of 850h. All other published results (using EM training and Bayesian inference) use context vectors as an additional source of information: This might be an explanation why Nuhn et al. (2012) and Ravi (2013) are able to outperform exact EM training as reported by Ravi and Knight (2011). (Ravi, 2013) reports the most efficient method so far: It only consumes about 3h of computation time. However, as mentioned before, those results are not directly comparable to our work, since they use additional context information on the target side.

Our algorithm clearly outperforms the exact EM training in run time, and even slighlty improves performance in BLEU. Similar to the simple substitution case, the improved performance might be caused by inferring a sparser distribution $p_{lex}(f|e)$. However, this requires further investigation.

5 Conclusion

We have shown a conceptually consistent and easy to implement EM based training method for decipherment that outperforms exact and beam search EM training for simple substitution ciphers and decipherment for machine translation, while reducing training time to a fraction of exact and beam EM. We also point out that the preselection method presented in this paper is not restricted to word based translation models and can also be applied to phrase based translation models.

References

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 (ACL)*, pages 1040–1047, Uppsala, Sweden, July. The Association for Computer Linguistics.

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, B*, 39.

Xuedong Huang, Fileno Alleva, Hsiao wuen Hon, Mei yuh Hwang, and Ronald Rosenfeld. 1992. The sphinx-ii speech recognition system: An overview. *Computer, Speech and Language*, 7:137–148.

Kevin Knight, Anish Nair, Nishit Rathod, and Kenji Yamada. 2006. Unsupervised Analysis for Decipherment Problems. In *Proceedings of the COLING/ACL on Main conference poster sessions*, COLING-ACL '06, pages 499–506, Stroudsburg, PA, USA. Association for Computational Linguistics.

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 (ACL)*, pages 156–164, Jeju, Republic of Korea, July. Association for Computational Linguistics.

Malte Nuhn, Julian Schamper, and Hermann Ney. 2013. Beam search for solving substitution ciphers. In *Annual Meeting of the Assoc. for Computational Linguistics*, pages 1569–1576, Sofia, Bulgaria, August.

Chris Pal, Charles Sutton, and Andrew McCallum. 2006. Sparse forward-backward using minimum divergence beams for fast training of conditional random fields. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Sujith Ravi and Kevin Knight. 2008. Attacking decipherment problems optimally with low-order n-gram models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 812–819, Honolulu, Hawaii. 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 (ACL-HLT)*, pages 12–21, Portland, Oregon, USA, June. Association for Computational Linguistics.

Sujith Ravi. 2013. Scalable decipherment for machine translation via hash sampling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 362–371, Sofia, Bulgaria, August. Association for Computational Linguistics.

Jörg Tiedemann. 2009. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria.

Wolfgang Wahlster, editor. 2000. *Verbmobil: Foundations of speech-to-speech translations*. Springer-Verlag, Berlin.

S.J. Young and Sj Young. 1994. The htk hidden markov model toolkit: Design and philosophy. *Entropic Cambridge Research Laboratory, Ltd*, 2:2–44.