Decipherment as Regression: Solving Historical Substitution Ciphers by Learning Symbol Recurrence Relations

Nishant Kambhatla **Logan Born** **Anoop Sarkar**
School of Computing Science, Simon Fraser University
8888 University Drive, Burnaby BC, Canada
{nkambhat, loborn, anoop}@sfu.ca

Abstract

Solving substitution ciphers involves mapping sequences of cipher symbols to fluent text in a target language. This has conventionally been formulated as a search problem, to find the decipherment key using a character-level language model to constrain the search space. This work instead frames decipherment as a sequence prediction task, using a Transformer-based causal language model to learn recurrences between characters in a ciphertext. We introduce a novel technique for transcribing arbitrary substitution ciphers into a common *recurrence encoding*. By leveraging this technique, we (i) create a large synthetic dataset of homophonic ciphers using random keys, and (ii) train a decipherment model that predicts the plaintext sequence given a recurrence-encoded ciphertext. Our method achieves strong results on synthetic 1:1 and homophonic ciphers, and cracks several real historic homophonic ciphers. Our analysis shows that the model learns recurrence relations between cipher symbols and recovers decipherment keys in its self-attention.[1]

1 Introduction

Text may be considered a special kind of recurrent sequence, where letters repeat at intervals which conform to a language's character n-gram distribution. The hidden mapping between cipher text and plain text can be viewed as a model that predicts this recurrent sequence. Can we use self-attention to recover the mapping from a sequence of recurrent symbols to crack the cipher?

In this work, we exploit this idea for decipherment by building upon recent successes of Transformer models (Vaswani et al., 2017) in reasoning-based regression tasks such as mathematical reasoning (Saxton et al., 2019; Li et al., 2021) and learning the mathematical function for recurrent sequences (D'Ascoli et al., 2022). We rethink decipherment as a regression task that predicts a natural language plaintext by learning a recurrence relation between integer-coded ciphertext symbols.

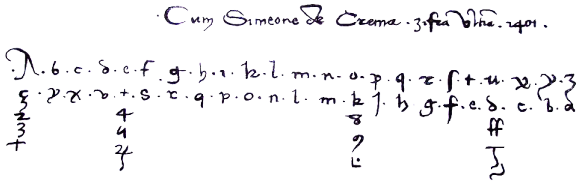


Figure 1: The homophonic substitution key for the *Simeone de Crema* written in Mantua in 1401 AD. The top line maps each character in the alphabet to its reversed-alphabet equivalent; each vowel is substituted by three additional symbols.

There exist large collections of historical ciphers (see de-crypt.org)[2], in the form of encrypted letters and more informal communications, of which many remain undeciphered. Many of these texts employ complex *homophonic substitution ciphers*, which mask the frequencies of letters by using a larger alphabet than the underlying language. Figure 1 shows the first known homophonic cipher from 1401 AD [3]. Automated computational decipherment of such texts is challenging (Pettersson and Megyesi, 2019; Megyesi et al., 2020). Prior work has mainly focused on using clever heuristics and/or search algorithms to explore the space of cipher keys and score multiple candidate plaintexts under character language models (LMs) (Knight et al., 2006; Corlett and Penn, 2010; Hauer et al., 2014; Berg-Kirkpatrick and Klein, 2013; Nuhn et al., 2013, 2014; Kambhatla et al., 2018) In contrast Aldarrab and May (2021) train a sequence-to-sequence model to solve simple (one-to-one) substitution ciphers. This approach, however, cannot solve complex homophonic ciphers as it relies on frequency information which such ciphers obscure.

In this paper, in a departure from frequency and

[1] https://github.com/protonish/decipher_symbol_recurrence

[2] https://de-crypt.org/decrypt-web/RecordsList

[3] https://en.wikipedia.org/wiki/Francesco_I_Gonzaga

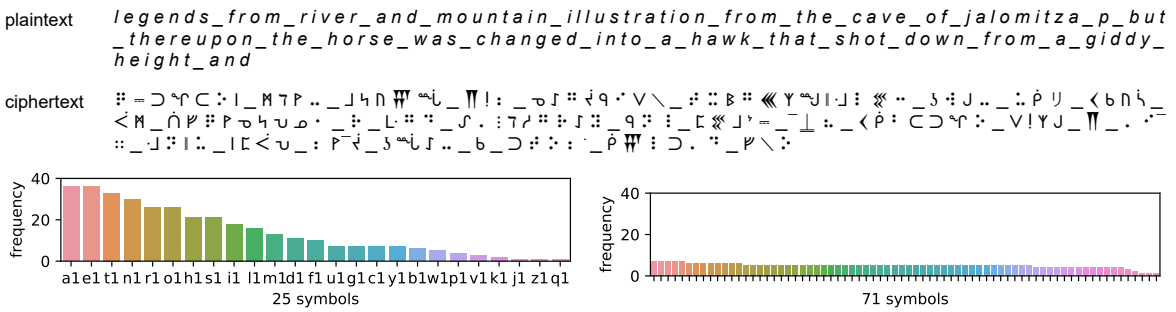


Figure 2: Part of an arbitrary 424 characters long homophonic cipher with 71 symbols and its plaintext. **Bottom** row juxtaposes symbol frequencies in 1:1 vs homophonic encipherments of the plaintext.

heuristic search-based techniques, we make the following **contributions**: (1) We create a sequence-to-sequence dataset comprising 2 million unique homophonic ciphers and their plaintext. (2) We propose a novel *recurrence encoding* which encodes information about the position and repetition of symbols in a cipher. This can be applied to both 1:1 and homophonic ciphers. (3) This encoding allows us to treat decipherment as a sequence prediction task conditioned on an integer sequence. We introduce a novel approach to solving homophonic ciphers by using a Transformer LM to translate integer-encoded ciphertexts to plaintexts. We also provide exhaustive analysis to show the strengths of our model. (4) We demonstrate near-perfect results on synthetic homophonic ciphers. Additionally, we show fully automated decipherment of `TNA_SP106/5` and `BnF-f01`, two real historical ciphers.

Our analysis shows that reproducing the input ciphertext before generating the plaintext helps our model to learn the relations between cipher symbols. This enables it to implicitly learn the decipherment key with high accuracy, and to determine which symbols are homophones of one another. The decipherment is highly constrained by this implicit key even in the face of disfluent plaintexts, and as a result our model is able to produce decipherments into Latin and late Middle/early modern English, despite being trained on modern English.

2 Decipherment of Substitution Ciphers

A 1:1 or *simple substitution cipher*, the oldest known technique for obscuring written information, defines a 1-1 mapping between plaintext characters and ciphertext symbols. This mapping can easily be broken with frequency analysis (Hauer et al., 2014; Kambhatla et al., 2018; Aldarrab and May, 2021) which leverages the fact that these ciphers preserve the distribution of character frequencies in the underlying language.

Method	Search	Train
❄ *n*-gram LM (2010)	A*	✗
❄ LM + Bay. Inf.(2011)	sampling	✗
❄ LM + HMM (2013)	EM; 1M restarts	✓
❄ *n*-gram LM (2013; 2014)	beam	✗
❄ lstm LM (2018)	beam	✗
🔥 Generative LM (Ours)	✗	✓

Table 1: Summary of different methods used for solving homophonic ciphers. Prior approaches are predominantly search-based and use a frozen language model to score partial candidate hypotheses.

Homophonic ciphers are substitution ciphers where one plaintext character may be encoded by more than one ciphertext symbol. In this way, frequent plaintext characters can be mapped to many infrequent ciphertext symbols, resulting in a flattened frequency distribution (Figure 2).

2.1 Background

Traditional approaches to natural language decipherment of homophonic substitution ciphers–the main focus of this work–are entirely search-based (Table 1). Nuhn et al. (2013) perform a beam search using an offline, frozen character language model to score candidate decipherments, and Nuhn et al. (2014) improve the rest-cost estimation for this technique. Kambhatla et al. (2018) further improve the rest-cost heuristic by using a frozen neural LM to score hypotheses. Corlett and Penn (2010), on the other hand, use A* search. Berg-Kirkpatrick and Klein 2013 uses 1 million random restarts to learn HMMs for decipherment.

The ability of such inference-only methods to generalize can be limited, as it depends on the underlying language model that is used to score the

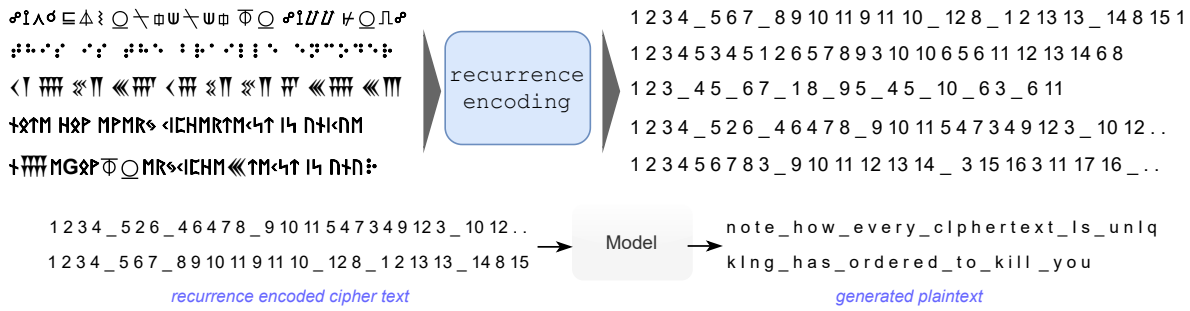


Figure 3: **Top-left**: Before *recurrence encoding*. Every ciphertext is unique with a different key, with distinct keys and plaintexts, and varying lengths. There is no relation between identical symbols in different ciphers (*e.g.* $\overline{\Phi}$ in cipher 1 and 5). These dissimilarities make it nearly impossible to train a model to generalize to new, unseen homophonic ciphers. **Top-right**: After *recurrence encoding*. Arbitrary ciphertexts are converted to recurrent integer sequences. The encoding is applied to each cipher independently: the same symbol may receive different encodings in different ciphertexts depending on where it first occurs, and two ciphers may receive the same encoding despite using different alphabets. **Bottom:** Recurrence-encoded symbols decipher to different values depending on context.

constructed hypotheses. Even an efficient search can take a long time to find the whole cipher key (Nuhn et al., 2013; Kambhatla et al., 2018).

3 Our Generative Decipherment Model

We frame decipherment as a novel sequence generation task–we train a generative language model that learns the relations between recurring symbols in a homophonic cipher and generates the corresponding deciphered plaintext message.

3.1 Converting Arbitrary Ciphers into Recurrent Integer Sequences

In a substitution cipher, any character may be substituted for any other, limiting what one can generalize between different ciphertexts (Figure 3). However, unrelated ciphers may still exhibit similar patterns of letter distribution and repetition and display latent characteristics of the plaintext language. For example, the characters at the beginning of two unrelated ciphers are likely drawn from the same distribution of word-initial letters in the underlying plaintext language. So we can generalize better by treating ciphers as recurrent sequences.

We propose a novel *recurrence encoding* to highlight where cipher symbols first occur and how they are repeated within a ciphertext. This encoding replaces the nth unique symbol in a ciphertext with the number n wherever that symbol occurs (Figure 3). This converts arbitrary ciphertexts into integer sequences, and thus provides a coherent connection between ciphers with distinct keys or disjoint alphabets.

3.2 Modelling Symbol Recurrence Relations

In this section, *ciphertext* specifically refers to a recurrence-encoded integer sequence.

Following Wang et al. (2021) and Zhang et al. (2022) in MT, we use a causal language model (LM) as a replacement for a Transformer-based encoder-decoder model (Vaswani et al., 2017). For a source sequence X and the target Y,

$$[X^l, Y^l] = \text{FFN} \circ \text{SelfAtn}\left([X^{l-1}, Y^{l-1}], Mask\right) \quad (1)$$

where l is the layer index, FFN is a feed-forward network, and $Mask$ denotes the attention mask. Our **CausalLM** model is a unidirectional LM, with causal masking over both the source and the target. This optimizes the joint distribution of cipher (src) and plaintext (tgt) sequences:

$$\begin{aligned}\mathcal{L}^{CLM}(X,Y) &= \mathcal{L}^{SRC} + \mathcal{L}^{TGT} \\ &= -\log P(X) - \log P(Y|X)\end{aligned} \quad (2)$$

CausalLM is therefore forced to sequentially predict the ciphertext just as it predicts the plaintext. This formulation encourages the model to learn a coherent relationship between the ciphertext and plaintext characters within each training sample.

Baseline Models. To understand the importance of causal attention masking, we also consider the following models which do not generate the ciphertext:

Seq2Seq Following (Aldarrab and May, 2021), this is a character level Transformer architecture that is only optimized on the target-side (plaintext) loss: $\mathcal{L}^{\text{Seq2Seq}}(X,Y) = \mathcal{L}^{TGT} = -\log P(Y|X)$

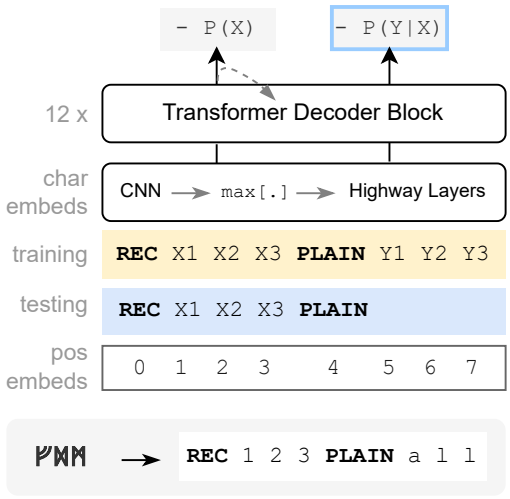


Figure 4: Schematic depiction of our Transformer LM model for an example ciphertext. `X` denotes the recurrence encoded ciphertext and `Y` is the plaintext, prepended by `REC` and `PLAIN` tags respectively.

Target-Only CausalLM is only optimized on the target-side loss $\mathcal{L}^{TGT}$, and incurs no loss when generating the source text.

PrefixLM combines SelfAtn and masked-SelfAtn: the ciphertext is attended at all times, while the plaintext uses a causal mask:

$$Mask(i,j) = 1, \text{ if } i \geq j \text{ or } j \leq |X|; \text{ else } 0 \quad (3)$$

where $1 \leq i,j \leq (|X| + |Y|)$. This setting mimics an encoder-decoder by modelling the conditional distribution of the plaintext target given the ciphertext source with target-only objective $\mathcal{L}^{PLM}(X,Y) = \mathcal{L}^{TGT}$.

All models build character embeddings with a convolutional neural network and highway networks over character inputs (Kim et al., 2016).

4 Experimental Setup

4.1 Data

Length	Keys	#Train	#Valid	#Test
300-400	30-45	460,467	25,582	25,581
	45-60	503,695	25,582	25,581
	30-85	542,611	25,582	25,581
300-700	30-45	460,467	25,582	25,581
	45-60	542,611	25,582	25,581
	30-85	1,046,306	25,582	25,581

Table 2: Summary of the synthetic homophonic ciphers used in our experiments. All ciphers are unique.

We extract 1000 English books from Project Gutenberg[4] to create training, validation and test sets. We also use ∼200k English sentences from `news-commentary v9` from WMT14 En-De. Combining these, we generate homophonic ciphers with lengths and keys summarized in Table 2.

Algorithm 1 Allocate Homophonic Symbols

Plaintext sample y of length n
Plaintext chars, y_{freq} = `Counter`(y).`most_common()`
Approx. cipher symbols, #sym

procedure HOMOPHONIC(y_{freq}, n, #sym)
  sym_count = 0 ▷ final num. of cipher symbols
  sym_per_char = `dict()` ▷ num. symbols per plain char
  for char, freq `in` y_{freq} **do**
    char_weight = `int`(freq / n)
    w_{sym} = `int`(char_weight * #sym)

$$\text{num\_sym} = \begin{cases} 1, & w_{sym} == 0 \\ w_{sym}, & \text{otherwise} \end{cases}$$

    sym_count += num_sym
    sym_per_char[char] = num_sym
  return sym_count, sym_per_char

Synthetic Homophonic Ciphers. To train a model that can generalize to unseen ciphers, we first generate synthetic homophonic ciphers using Algorithm 1 to flatten the frequency distribution of a text. This technique allocates multiple less-frequent ciphertext symbols to common plaintext letters to yield strong homophonic ciphers.

For simple substitution ciphers, we use the same English data as above to create 1.2M synthetic substitution ciphers with lengths up to 256. Following previous work on 1:1 ciphers (Nuhn et al., 2013; Kambhatla et al., 2018; Aldarrab and May, 2021), we evaluate on 50 test ciphers of lengths up to 128 (16,32,64) and beyond 128 (128,256) from the Wikipedia page on History [5]. All our experimental settings include data with word boundaries denoted by the space symbol (_). We train our multilingual model on length 256 1:1 ciphers from the 13 language data in Aldarrab and May (2021)[6] which includes training, validation and test splits.

4.2 Model Details

Our main model uses a Transformer decoder-based auto-regressive language model. Our model comprises a 12 layer decoder with 12 attention heads and a feed-forward dim. of 1536, totalling 23M trainable parameters. We use character filters of

[4] https://github.com/pgcorpus/gutenberg
[5] https://en.wikipedia.org/wiki/History
[6] https://github.com/NadaAldarrab/s2s-decipherment

$[(1, 64), (2, 128), (3, 192), (4, 256)]$ with a character dim of 4 and 2 highway layers. For the seq2seq model, we implement a 6 layer encoder-decoder Transformer with the same settings as above. All models are implemented using the `fairseq` toolkit (Ott et al., 2019).

All models train for about 30 iterations over the data at ∼100 minutes per epoch on 4xA6000 GPUs. Inference uses a beam size of 200 unless otherwise stated. Inference speed is about 400 chars/second on a single Titan RTX.

Evaluation Following prior work (Kambhatla et al., 2018; Aldarrab and May, 2021), we evaluate on Symbol Error Rate (SER), the proportion of ciphertext symbols which are wrongly recovered.

5 Homophonic Substitution Ciphers

We experiment on solving our own synthetic homophonic ciphers, and on automatically deciphering two real world homophonic ciphers that have previously only been cracked manually.

5.1 Results on Synthetic Ciphers

Table 3 reports results on synthetic homophonic data using recurrence encoding. On 400- and 700-character long ciphers, our best model with causal attention and up to 65 keys achieves near perfect decipherment. As expected, we observe the best results on long ciphers, which provide more context. Even on challenging ciphers with up to 85 keys over 400 characters (4.5 chars/symbol), our model attains an average error rate of 2.25%, averaging only 1 wrong character each. As will be shown in Section 8 (Table 7), our model also implicitly recovers decipherment keys with $> 98\%$ accuracy.

#keys	Model	Max Len. 400	Max Len. 700
30-45	Seq-to-Seq	72.30	`fail`
	PrefixLM	54.73	69.50
	CausalLM (tgt)	29.99	37.20
	CausalLM	**0.40**	**0.21**
40-65	PrefixLM	69.50	54.73
	CausalLM (tgt)	29.99	37.20
	CausalLM	**0.83**	**0.80**
30-85	PrefixLM	70.52	71.82
	CausalLM (tgt)	42.05	42.69
	CausalLM	**2.25**	**2.19**

Table 3: SER% on synthetic, homophonic, recurrence-encoded ciphers. All plaintexts use 26 unique letters.

These results show the strength of recurrence encoding together with a causal LM objective.

Generating cipher + plaintext vs. plaintext only: The best results by a significant margin are obtained through language modeling with a causal attention mask—the only model that is trained to generate the ciphertext before the intended plaintext. All other approaches fail to give adequate results in any setting. A sequence-to-sequence model obtains poor results on ciphers of 400 characters and fails to converge on 700 char long ciphers.[7] Prefix attention is consistently worse than causal attention, and its performance varies unpredictably across input length and number of keys. Causal attention with target-only loss is worse than causal attention, suggesting that reproducing the source text may play an important role in solving this task. We consider this idea further in our analysis of the model's attention in Section 8.

5.2 Results on Zodiac 408 Cipher

We compare our method on the famous Zodiac 408 cipher that has 408 characters written with 54 different symbols (~7.5 characters per symbol). For this particular cipher, all models including ours are trained on the English Gigaword corpus for fair comparison with other models in Table 4. From Table 4, our generative LM is both better, and faster by orders of magnitude.

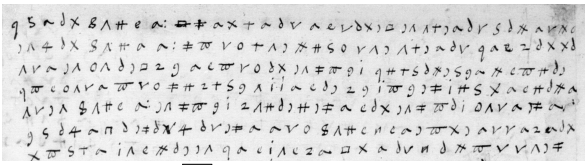
Method	Search	SER (%)	Speed
LM+EM (2013)	1M restarts	11.0	–
$n-$gram LM (2013)	beam 100K	94.6	4000
	beam 1M	2.7	35000
LSTM LM (2018)	beam 100K	2.4	5600
	beam 1M	1.9	50000
Ours (greedy)	beam 1	**1.9**	1 sec
Ours (best)	beam 200	**1.9**	2 sec

Table 4: Zodiac 408. Methods for simple (1:1) substitution ciphers can not be used on Zodiac408. The last column shows inference speed in seconds. Our method is much faster because it auto-regressively generates the decipherment whereas previous methods perform an exhaustive search to find the mapping for each symbol.

5.3 Solving historical substitution ciphers

Most historical ciphers are centuries old and can be challenging to solve—the encipherment scheme may be peculiar to the author; the language may be

[7] NMT models are known to suffer from long sentences (Neishi and Yoshinaga, 2019; Varis and Bojar, 2021)



p e i s _ j o u r _ h e s _ b e i n _ e r
n i s t _ t o _ o b t e i n _ l i c e n c e _ t
o _ k i s _ j o u e _ h a n d _ b o t _ c u l
d _ n o t _ o b t e i n _ p e r m i s s i o n e
_ t o _ d o _ i t _ m y _ e r a n d _ i s _ t o
_ h a y f _ p u b l i c t l y _ e c r a u i t _
p a r d o n e _ a n d _ h u m b l y _ o f f e r
_ i t _ m y _ f a y t h f u l

p l e i s _ y o u r _ h e s _ b e i n _ e r n i
s t _ t o _ o b t e i n _ l i c e n c e _ t o _
k i s _ y o u e _ h a n d _ b o t _ c u l d _ n
o t _ o b t e i n _ p e r m i s s i o n e _ t o
_ d o _ i t _ m y _ e r a n d _ i s _ t o _ h a
y f _ p u b l i c t l y _ e c r a u i t _ p a r
d o n e _ a n d _ h u m b l y _ o f f e r _ i t
_ m y _ f a y t h f u l

Figure 5: Predicted (middle) vs true (bottom) decipherment of the first few lines of BnF-f01 cipher (top). Errors in red and boxed.

archaic or unstandardised, and thus out-of-domain for models trained on modern data; or there may be human errors in the transcription.

Though our model is trained only on synthetic ciphers with no further finetuning, we hypothesize that it may nonetheless see success on real medieval ciphers. This section demonstrates those successes. **1. TNA_SP106/5** Also called `CharlesI_(0096)`,[8] this is a strong homophonic cipher that was written in 1624 in the United Kingdom, during the reign of King Charles I of England. It is a very difficult cipher which is 171 characters long, using 47 unique symbols to encipher 27 plaintext letters (each cipher symbol appears 3.6 times on average).

Using our homophonic 40-65 key model with beam size 1000, we attain a readable decipherment with a symbol error rate **(SER) 18%**.

2. BnF_fr2988_f01 `BnF-f01`[9] (Fig. 5) is a 2 page long enciphered letter from between 1524–1549 in Italy, believed to be addressed to King Henry VIII. This homophonic cipher uses 35 symbols, with archaic spellings in the underlying plaintext that make it a challenging target due to the different character distribution from our training set. Examples include `pleis`→(please), `faythful`→(faithful), `obtein`→(obtain) and `gretast`→(greatest). Our best homophonic model trained with ciphers between [30-45] keys is able to crack it with **SER 1.13%**. The first few lines of model output and corresponding plaintext are shown in Figure 5. Although our model was never explicitly trained for robustness, the recurrence encoding helps it to overcome the unexpected plaintext distribution and maintain a consistent key to recover the message.

6 Does our technique generalize to 1:1 substitution ciphers?

To exploit the well-known weakness of simple substituion ciphers, Aldarrab and May (2021) proposed *frequency ranking* whereby cipher symbols are replaced by their frequency ranks across all 1-1 substitution ciphers. We use recurrence encoding and frequency ranking with our best performing causal LM architecture[10] and compare with several baselines, including Aldarrab and May (2021).

cipher length →	**<128**	**>128**
Beam + 6-gram (Nuhn et al., 2013)	22.00	0.00
Beam + LM ((Kambhatla et al., 2018))	10.89	0.00
Beam + LM + Freq. Match (ibid.)	11.32	0.00
Seq2Seq + Freq. (Aldarrab and May)	7.68	0.00
Causal LM + Freq.	10.56	0.00
Causal LM + Rec.	11.30	0.02

Table 5: On simple substitution ciphers of length >128, our performance equals or exceeds all baselines. `Freq.` and `Rec.` denote frequency and recurrence encodings.

Our model performs well on simple substitutions (Table 6) using frequency ranking. While the scores on very short ciphers (16, 32, 64) only match the performance of beam-search based methods, on ciphers longer than 128, our model achieves close to 0 SER. Recurrence encoding is less effective in shorter sequences (< 128) and requires more context to be effective compared to frequency ranks which more directly indicate the plaintext character distribution. However, recurrence encoding is not required in this context as the character distributions are not flattened.

7 Unknown Plaintext Language

As Megyesi et al. (2020) reports, several historical ciphers in libraries and archives have no information on the plaintext language. We evaluate on the

[8] de-crypt.org/decrypt-web/RecordsView/420
[9] de-crypt.org/decrypt-web/RecordsView/2323

[10] Recall that recurrence encoding doesn't work well with an encoder-decoder model (Table 3).

Latin Borg Cipher[11], a ca. 17th century manuscript, to learn if our model can decipher without explicit knowledge of the underlying language.

7.1 Multilingual model with no language ID

Following Aldarrab and May (2021), we train our decipherment model on ciphers from 13 different languages without language ID and apply it to the Borg cipher (page 0011v). Both models are trained on frequency based encoding. Compared to 5.47% SER in the baseline, our model achieves a better SER of 4.1%.

	SER (%)
Multilingual Seq2Seq (2021)	5.47
Multilingual Causal LM (ours)	**4.10**

Table 6: Using our multilingual model trained on 13 languages to solve the 1:1 `Borg` cipher.

7.2 Zero-shot decipherment in an unseen language

Though the Borg MS uses a simple substitution cipher, the plaintext language (Latin) is out-of-domain for our main model which was trained on English only (Sec. 5). Zero-shot inference on the first 400 characters of page 0011v results using our recurrence-encoding based model in an SER of 45.14%. A mere 3 manual inteventions–fixing two words (*aperitione*, *emorrhoidarum*) and correcting one (*cumo* to *cum*)–however, are sufficient to solve the entire cipher with **SER 3.89%**. See Appendix B for details on human-in-the-loop decipherment.

This shows that our model learns to consistently produce the same output for a given symbol regardless of the plaintext distribution, which is crucial for cracking ciphers and separates this from a conventional text generation model.

8 Analysis

8.1 On the significance of learning the distribution of ciphertext characters

Section 5 demonstrated that CausalLM significantly outperforms other models on synthetic homophonic data. This is the only approach which must model the distribution of ciphertext characters: Seq2Seq and PrefixLM allow the model to freely attend to the full ciphertext, while target-only loss gives no penalty for mistakes in the ciphertext. These settings remove the incentive to learn the distribution of ciphertext characters, whereas CausalLM must sequentially predict the ciphertext just as it predicts the plaintext.

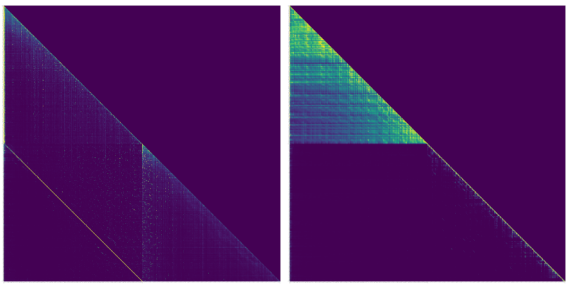


Figure 6: **Left:** self-attention map from our causal LM. **Right:** self-attention map over the same sentence from our causal LM with target-only loss.

Figure 6 illustrates the impact of target-only loss by showing the final layer of self-attention scores for an example input. CausalLM exhibits a strong diagonal pattern in the lower-left of the attention matrix, showing that it attends monotonically to cipher symbols when producing corresponding plaintext symbols. With target-only loss, attention is roughly uniform over all ciphertext symbols when reproducing the input, as there is no penalty for mistakes in this section and consequently no need to attend to relevant context cues. This model does not strongly attend to the ciphertext at any point when generating the plaintext.

Key Recovery The model's self-attention implicitly recovers decipherment keys. We construct a `n_plaintext_symbols` by `n_cipher_symbols` matrix where cell (p, c) sums the mean attention over all layers paid to c when producing p. More common symbols receive more attention, so we divide each column by the frequency of the corresponding symbol. Figure 7 depicts such a matrix, normalized so that the largest value in each row is 1: the largest value in most columns clearly corresponds to the decipherment key. Table 7 reports key error rate (KER) using this technique, as well as variants without adjusting for frequency or normal-

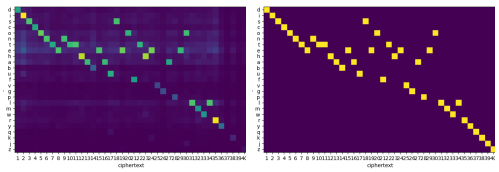


Figure 7: **Left:** Avg. attention paid to ciphertext symbols when generating plaintext symbols in one test case. **Right:** True key, where a dark cell indicates which character the symbol in each column deciphers to.

[11] https://cl.lingfil.uu.se/~bea/borg/

	KER (%)
Row-norm.	3.40
Col-norm.	3.63
Row-norm. + Frq. adj.	**1.54**
Col-norm. + Frq. adj.	3.63

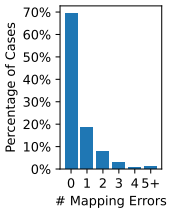


Table 7: (l) Key error rate (%) averaged over 2000 homophonic test cases. Values multiplied by 100 for readability. (r) Distribution of mapping errors.

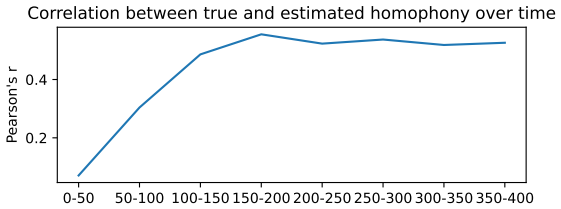


Figure 9: Pearson's r between self-attention scores at different time-steps and reference matrices encoding homophones (cf. Figure 8). Averaged over 50 ciphertexts.

izing by column rather than by row; the distribution of errors is shown beside the table. With our best technique, the median and mode number of erroneous mappings is 0, i.e. we perfectly recover the key from the self-attention in most cases.

Our model's ability to accurately produce Latin highlights that it learns to obey the inferred key even in the face of conflicting signals from an out-of-domain plaintext.

8.2 Recovering character recurrence relations

Homophone Recovery Attention from ciphertext symbols to other ciphertext symbols reveals which characters are homophonic. Figure 8 (left) shows, for a sample input, the average self-attention from a ciphertext symbol towards other ciphertext symbols; Figure 8 (right) shows which of these symbols are homophones. The largest self-attention scores roughly correlate with cells representing homophonic symbol pairs. Comparing to Figure 7 (left) we see that the homophone pairs which are *not* recovered involve those symbols for which the model lacks a confident plaintext mapping.

This behaviour arises as the model reproduces the input ciphertext. This can be observed by averaging self-attention over chunks of successive time-steps (rather than the entire input) and measuring Pearson's r between the resulting matrices and a reference matrix as in Figure 8 (right). As seen in Figure 9, the correlation between homophony and self-attention grows steadily as the model reproduces the input, reaching a plateau after ∼200 tokens. (See also Figure 13 in Appendix C.) This demonstrates the importance of CausalLM, as it shows that the model makes crucial inferences about its input as it reproduces that input.

We emphasize that our model learns homophony relations and recovers decipherment keys *implicitly*, in just a single pass over the input ciphertext. Prior search-based techniques required a search over many candidate plaintexts in order to recover these same relations explicitly.

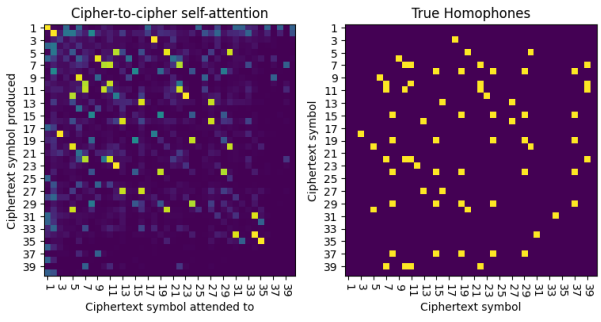


Figure 8: **Left:** Avg. attention from ciphertext symbols to other ciphertext symbols when reproducing one test case. **Right:** Dark cells indicate homophonous symbols.

9 Other Related Work

Computational decipherment based techniques have seen a wide range of applications ranging such as identifying unknown languages and scripts (Hauer and Kondrak, 2016), writing systems (Born et al., 2019, 2021, 2022) and lost languages (Snyder et al., 2010; Luo et al., 2019), offensive langauge detection (Wu et al., 2018; Qian et al., 2019), and, more recently, towards improving neural machine translation (Kambhatla et al., 2022). While decipherment has strong connections to cryptography research, we limit the scope of this work to natural language based decipherment. Knight et al. (2006) proposed an unsupervised noisy channel based technique for decipherment. Hauer et al. 2014 solved short ciphers with Monte-Carlo tree search. Greydanus (2017) train a seq-to-seq LSTM to solve polyalphabetic substitution ciphers including Enigma, but only explore supervised known-plaintext attacks. CipherGAN (Gomez et al., 2018) exploits learned letter embedding distributions, but requires a large volume of ciphertext and only handles 1:1 substitution and Vigenère ciphers. Luo et al. (2021) and Aldarrab and May (2022) pro-

pose techniques to decipher undersegmented ciphers. Aldarrab and May (2021) train a sequence-to-sequence neural translation model to decipher from character frequencies. In contrast, we introduce a novel encoding which is suitable for homophonic inputs, and demonstrate that a causal LM is more effective than a seq-to-seq model in a homophonic setting.

Cross-attention from encoder-decoder architectures has been shown to have limited explanatory power in translation settings (Moradi et al., 2019, 2021). In spite of this, Born et al. 2022 show that encoder self-attention implicitly captures information which replicates expert intuitions about document structure in an undeciphered script. In a similar vein, our key recovery experiments offer evidence of yet another way self-attention may be fruitfully exploited in the decoder in a decipherment setting.

10 Conclusion and Future Work

We introduce a novel recurrence encoding to represent distributional information which is invariant across plaintexts and ciphertexts, even under homophonic ciphers. This allows us to train a Transformer LM for decipherment using synthetic ciphertext-plaintext pairs. Our model achieves strong results on unseen homophonic substitution ciphers, and achieves the first fully-automatic decipherment of several historical ciphers. We show that language models vastly outperform sequence-to-sequence models on this task, and that causal attention masking (which forces our model to reproduce the ciphertext before deciphering it) is crucial to solve homophonic inputs. Our analysis shows that our model implicitly learns homophony relations and the decipherment key while reproducing the input. In a zero-shot setting, our model accurately deciphers into Latin despite being trained on English; This work marks a successful departure from search-based solutions to homophonic substitution ciphers, and introduces language models as a viable tool for future decipherment work.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. The research was partially supported by the Natural Sciences and Engineering Research Council of Canada grants NSERC RGPIN-2018-06437 and RGPAS-2018-522574 and a Department of National Defence (DND) and NSERC grant DGDND-2018-00025 to the third author, and by an NSERC award CGSD3-547773-2020 to the second author.

Limitations

A key limitation of our model is the combined sequence length of the ciphertext and the plaintext. As the standard self-attention mechanism of the Transformer uses $O(n^2)$ time and space with respect to sequence length, modelling longer ciphers (eg. 1500 chars) is extremely compute inefficient. This also restricts our model's ability to handle ciphers such as the Beale Pt. 2 cipher. A possible avenue for an extension of this work is to address this issue by leveraging more sophisticated self attention mechanisms like the linformer (Wang et al., 2020).

Ethics Statement

This work is concerned with decoding encrypted correspondences, and therefore the techniques in the paper are designed to reveal information that has been purposefully obscured and might violate the privacy of the authors. However, an encryption system such as the homophonic substituion cipher is primarily seen in centuries old historical ciphers, and is both relatively weak and obselete. The methods might have little impact beyond any applications intended towards decipherment of ancient ciphers or machine translation. Further, the more standard encryption techniques such as the AES/RSA are very sophisticated and cannot be attacked with the model discussed in the paper.

We note that our proposal of this method is not as a replacement to expert code-breakers, but as a new tool at their disposal. Our model cannot "cheat" except by disobeying the key, and we've shown that it does consistently follow the key (Section 8.2). Thus our model output is no more or less trustworthy than the equivalent produced by a human. Since there is no guarantee that the model will always produce the right decipherment, it is imperative that domain experts assess the text produced by this model in the same way they would assess proposals from an amateur decipherer with little/no domain knowledge.

References

Nada Aldarrab and Jonathan May. 2021. Can sequence-to-sequence models crack substitution ciphers? In

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 7226–7235, Online. Association for Computational Linguistics.

Nada Aldarrab and Jonathan May. 2022. Segmenting numerical substitution ciphers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 706–714, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Taylor Berg-Kirkpatrick and Dan Klein. 2013. Decipherment with a million random restarts. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 874–878, Seattle, Washington, USA. Association for Computational Linguistics.

Logan Born, Kate Kelley, Nishant Kambhatla, Carolyn Chen, and Anoop Sarkar. 2019. Sign clustering and topic extraction in Proto-Elamite. In *Proceedings of the 3rd Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 122–132, Minneapolis, USA. Association for Computational Linguistics.

Logan Born, Kathryn Kelley, M. Willis Monroe, and Anoop Sarkar. 2021. Compositionality of complex graphemes in the undeciphered Proto-Elamite script using image and text embedding models. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4136–4146, Online. Association for Computational Linguistics.

Logan Born, M. Monroe, Kathryn Kelley, and Anoop Sarkar. 2022. Sequence models for document structure identification in an undeciphered script. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9111–9121, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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, Uppsala, Sweden. Association for Computational Linguistics.

Stéphane D'Ascoli, Pierre-Alexandre Kamienny, Guillaume Lample, and Francois Charton. 2022. Deep symbolic regression for recurrence prediction. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 4520–4536. PMLR.

William F. Friedman. 1922. The index of coincidence and its applications in cryptography. In *Department of Ciphers. Publ 22.*, Geneva, Illinois. Riverbank Laboratories.

Aidan N. Gomez, Sicong Huang, Ivan Zhang, Bryan M. Li, Muhammad Osama, and Lukasz Kaiser. 2018. Unsupervised cipher cracking using discrete gans.

Sam Greydanus. 2017. Learning the enigma with recurrent neural networks.

Bradley Hauer, Ryan Hayward, and Grzegorz Kondrak. 2014. Solving substitution ciphers with combined language models. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2314–2325, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Bradley Hauer and Grzegorz Kondrak. 2016. Decoding anagrammed texts written in an unknown language and script. *Transactions of the Association for Computational Linguistics*, 4:75–86.

Nishant Kambhatla, Logan Born, and Anoop Sarkar. 2022. CipherDAug: Ciphertext based data augmentation for neural machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 201–218, Dublin, Ireland. Association for Computational Linguistics.

Nishant Kambhatla, Anahita Mansouri Bigvand, and Anoop Sarkar. 2018. Decipherment of substitution ciphers with neural language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 869–874, Brussels, Belgium. Association for Computational Linguistics.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush. 2016. Character-aware neural language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1).

Kevin Knight, Beáta Megyesi, and Christiane Schaefer. 2011. The copiale cipher. In *Proceedings of the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web*, pages 2–9, Portland, Oregon. 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*, pages 499–506, Sydney, Australia. Association for Computational Linguistics.

Wenda Li, Lei Yu, Yuhuai Wu, and Lawrence Charles Paulson. 2021. Isarstep: a benchmark for high-level mathematical reasoning. In *ICLR*.

Jiaming Luo, Yuan Cao, and Regina Barzilay. 2019. Neural decipherment via minimum-cost flow: From Ugaritic to Linear B. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3146–3155, Florence, Italy. Association for Computational Linguistics.

Jiaming Luo, Frederik Hartmann, Enrico Santus, Regina Barzilay, and Yuan Cao. 2021. Deciphering Undersegmented Ancient Scripts Using Phonetic Prior. *Transactions of the Association for Computational Linguistics*, 9:69–81.

Beáta Megyesi, Bernhard Esslinger, Alicia Fornés, Nils Kopal, Benedek Láng, George Lasry, Karl de Leeuw, Eva Pettersson, Arno Wacker, and Michelle Waldispühl. 2020. Decryption of historical manuscripts: the decrypt project. *Cryptologia*, 44(6):545–559.

Pooya Moradi, Nishant Kambhatla, and Anoop Sarkar. 2019. Interrogating the explanatory power of attention in neural machine translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 221–230, Hong Kong. Association for Computational Linguistics.

Pooya Moradi, Nishant Kambhatla, and Anoop Sarkar. 2021. Measuring and improving faithfulness of attention in neural machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2791–2802, Online. Association for Computational Linguistics.

Masato Neishi and Naoki Yoshinaga. 2019. On the relation between position information and sentence length in neural machine translation. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 328–338, Hong Kong, China. 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 (Volume 1: Long Papers)*, pages 1568–1576, Sofia, Bulgaria. Association for Computational Linguistics.

Malte Nuhn, Julian Schamper, and Hermann Ney. 2014. Improved decipherment of homophonic ciphers. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1764–1768, Doha, Qatar. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Eva Pettersson and Beata Megyesi. 2019. Matching keys and encrypted manuscripts. In *Proceedings of the 22nd Nordic Conference on Computational Linguistics*, pages 253–261, Turku, Finland. Linköping University Electronic Press.

Jing Qian, Mai ElSherief, Elizabeth Belding, and William Yang Wang. 2019. Learning to decipher hate symbols. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3006–3015, Minneapolis, Minnesota. Association for Computational Linguistics.

Sujith Ravi and Kevin Knight. 2011. Bayesian inference for zodiac and other homophonic ciphers. In *ACL*, pages 239–247.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2019. Analysing mathematical reasoning abilities of neural models. In *International Conference on Learning Representations*.

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, Uppsala, Sweden. Association for Computational Linguistics.

Dusan Varis and Ondřej Bojar. 2021. Sequence length is a domain: Length-based overfitting in transformer models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8246–8257, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Shuo Wang, Zhaopeng Tu, Zhixing Tan, Wenxuan Wang, Maosong Sun, and Yang Liu. 2021. Language models are good translators.

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Zhelun Wu, Nishant Kambhatla, and Anoop Sarkar. 2018. Decipherment for adversarial offensive language detection. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 149–159, Brussels, Belgium. Association for Computational Linguistics.

Biao Zhang, Behrooz Ghorbani, Ankur Bapna, Yong Cheng, Xavier Garcia, Jonathan Shen, and Orhan Firat. 2022. Examining scaling and transfer of language model architectures for machine translation.

A Hyperparameters and Settings

Preprocessing Following the previous work (Kambhatla et al., 2018; Aldarrab and May, 2021), we preprocessed the Project Gutenberg data by stripping the text of all non text elements, then lower-casing all characters, and removing all non-alphabetic and non-space characters. Our final plaintexts consists of the 26 letters of English alpahbet and the _ symbol to denote space only.

Multilingual Data. The 13 language multilingual data[12] released by Aldarrab and May (2021) consists of 2.2M ciphers in Catalan, Danish, Dutch, Finnish, French, German, Hungarian, Italian, Latin, Norwegian, Portuguese, Spanish, and Swedish languages.

Layers	12
Attn Heads	12
FF Dim.	1536
Char Embed	4
Highway Layers	2
Dropout	0.1
Attn. Dropout	0.1
Batch Size	32000
Peak lr	0.0005
Early Stopping	No
Max Epoch	20

Table 8: The hyperparameters for our model and training.

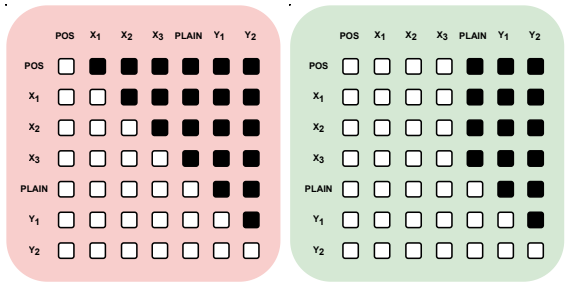


Figure 10: A high level depiction of the causal (left) and the prefix (right) attention masks. X denotes the ciphertext and Y is the plaintext, both prepended by POS and PLAIN tags respectively.

B Towards Decipherment with Human Intervention

In a realistic decipherment setting, the nature of the cipher under attack will not be known. It is possible that some symbols will be polyphonic, that the plaintext language will be out-of-domain, or that the text will be short or corrupted (intentionally, to prevent decipherment, or as a result of damage in the case of historical documents). In such cases, the outputs from an automated decipherment may require manual emendation; real computer-assisted decipherments have previously relied on post-editing by domain experts, as in the decipherment of the Copiale cipher Knight et al. 2011. We demonstrate two examples of how our model can be used for human-in-the-loop decipherment in this more realistic setting.

Zodiac 408 A famous cipher from the Zodiac killer of the 1970s, this text contains 408 characters written with 54 different symbols (~7.5 characters per symbol). There are six polyphonic[13] symbols, making the text out-of-domain for our model which was only trained on homophonic ciphers. Table 9 shows an example of assisted decipherment based on corrected words. When a correction is identified, it can be appended to the model input, for example:

```
orig. input:   REC <cipher> PLAIN
user prompt:   REC <cipher> PLAIN i _ l i k e
```

Since our model is a left-to-right language model on both cipher and plaintext, we can interrupt it at any point during plaintext generation to introduce a correction, which is then used as a new constraint on decoding the plaintext. Correcting only 5 polyphonous characters gives **SER 0.4%**, establishing a new state of the art on this cipher.

Borg Cipher The Borg Cipher[14] is a ca. 17th century manuscript written in enciphered Latin. Though the text uses a simple substitution cipher, the plaintext language is out-of-domain for our model which was trained on English.

Since our model was never trained on Latin, zero-shot inference on the first 400 characters results in an SER of 45.14%. But fixing the words `aperitione _ emorrhoidarum` and correcting

[12] https://github.com/NadaAldarrab/s2s-decipherment

[13] Different from a homophonic symbol, a *polyphonic* cipher symbol encodes more than one plaintext character.

[14] https://cl.lingfil.uu.se/~bea/borg/

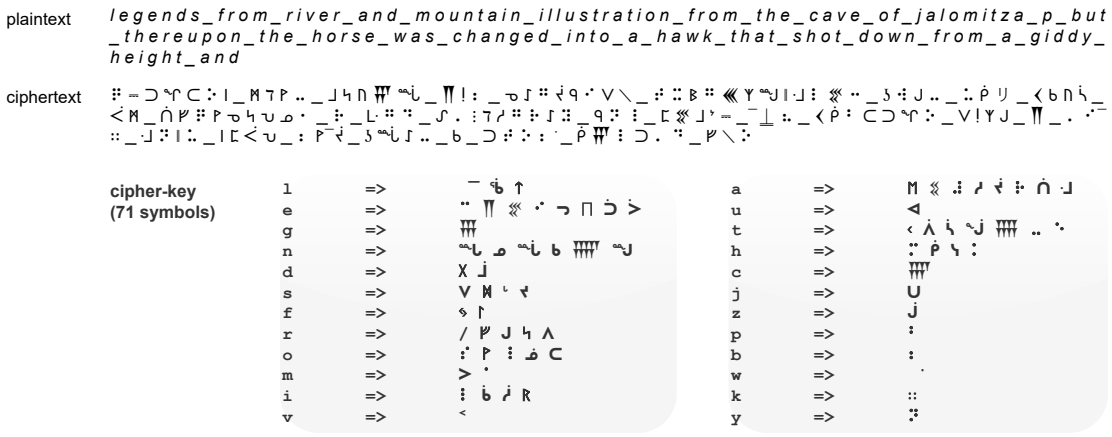


Figure 11: Example homophonic cipher and its substitution key showing the different symbols mapped to the plaintext characters.

Deciphered Text	Next Correction	SER
i_like_kadlang_peopde_because_it_in_so_much_fun_at_ax_more_fun_than_kilding_walz	people, is, killing (only one letter change in each: d → l, n → s, d → l)	14.40
i_like_kallang_people_because_it_is_so_much_fun_at_ax_more_fun_than_killing_walz	killing (a → i)	12.16
i_like_killing_people_because_it_is_so_much_fun_at_ax_more_fun_than_killing_wild	(no correction needed; partial key has been derived by the model)	3.04

Table 9: First 2 steps of the iterative human assisted decipherment of Zodiac-408 cipher. Identified corrections can be passed as the model input following the cipher. Fixing errors in the beginning ('killing') can lead to improved plaintext selections for symbols that appear later on ('wild'). After 3 more steps the process achieves SER of 0.4%.

cumo to _cum_ mostly solves the cipher in 3 steps resulting in **SER 3.89%**.

C Additional Analysis

Measures of Difficulty Figure 12 plots, for a sample of our test set, SER versus three measures of cipher difficulty: the index of coincidence (Friedman 1922, which measures the uniformity of the frequency distribution; values closer to 1 are more uniform and thus more challenging), the maximum number of homophones any ciphertext symbol has (where more homophones make a more challenging cipher), and the length of the cipher (where shorter ciphers are more ambiguous and provide less context). There is no significant correlation between SER and any of these three measures.

Attention Heads Figure 14 shows that each attention head demonstrates a consistent behaviour across multiple different inputs. (Full-size figure available in supplemental material.) Each column of this figure comes from a distinct ciphertext, and each row represents a Transformer layer, with the output layer on the bottom. The cells in this grid are divided into 12 sub-figures, each showing the self-attention map from one head in the corresponding layer on the corresponding input.

There are more heads which attend to the ciphertext in lower layers than in higher ones, suggesting that the model learns necessary features of the input early on. We note the presence of several heads which attend *near* the beginning of the ciphertext, but not to the very first tokens. This may reflect the fact that the first few tokens are always sequentially encoded as 1 2 3 ..., and that the model is focusing on the part of the text where these symbols first *repeat* rather than where they first occur.

Several of the attention maps exhibit clear vertical lines, meaning the head is attending to the same token(s) (often in the plaintext) at all subsequent time-steps. We speculate that these tokens may convey some crucial distributional information which helps to establish the key.

D Real Ciphers Used in This Work

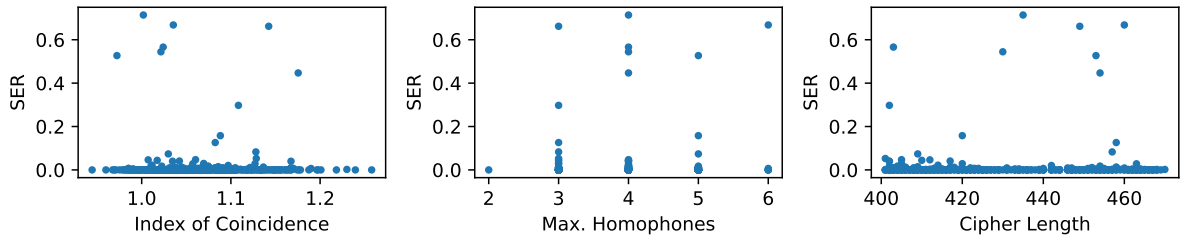


Figure 12: SER vs. index of coincidence, number of homophones of the most homophonic ciphertext symbol, and cipher length. SER is not significantly correlated with any of these metrics.

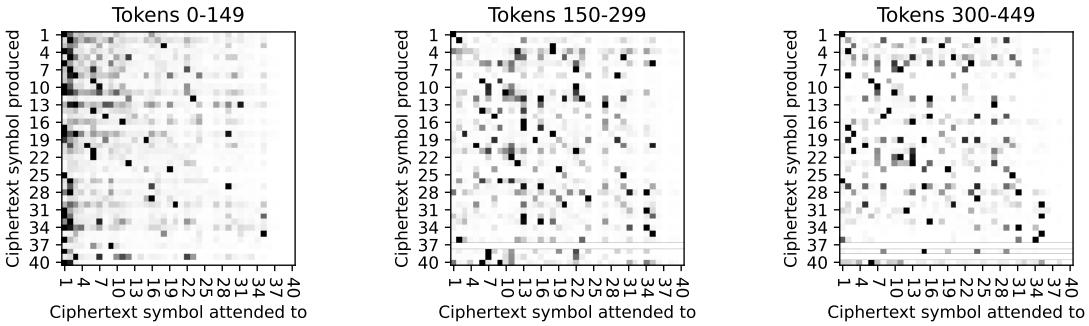


Figure 13: Mean self-attention from each ciphertext symbol to every other ciphertext symbol, averaged across different time-steps. This figure uses the same input as Figure 8. Note how the left subfigure, representing the earliest time-steps, does not meaningfully resemble the reference matrix from Figure 8, implying that the model has not learned which tokens are homophones at this early stage.

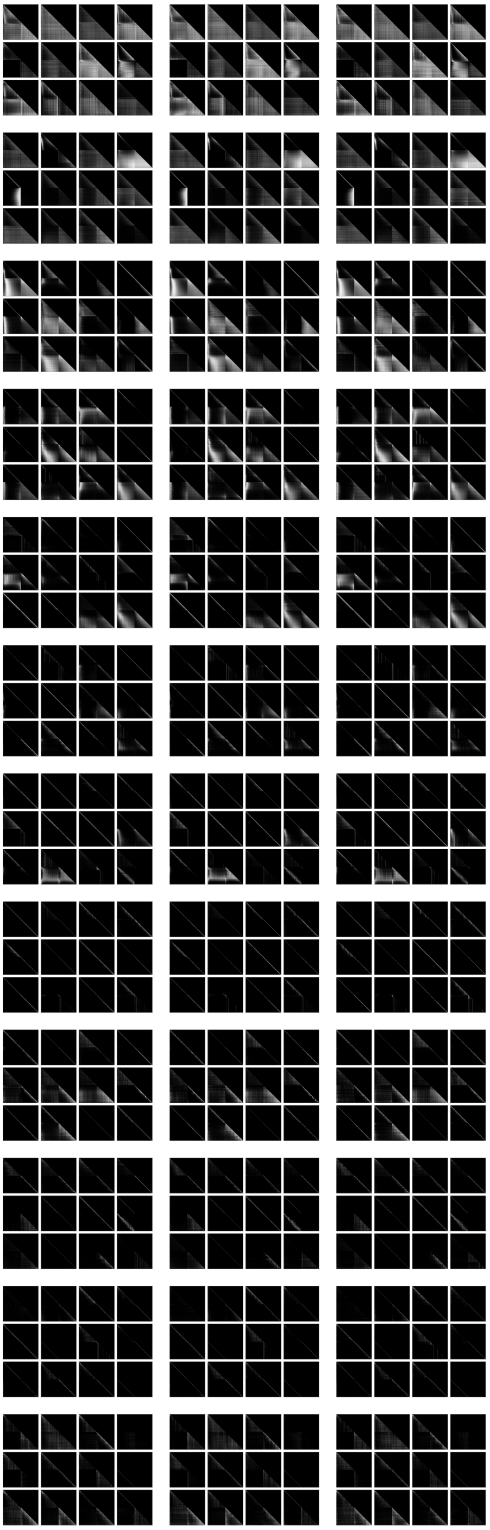


Figure 14: Attention maps for each head in each layer on three sample inputs. (Full-size figure available in supplemental material.) Each column represents a distinct ciphertext, and each row a Transformer layer, with the output layer on the bottom. Each cell is divided into 12 sub-figures, showing the self-attention maps from each of the 12 heads in the corresponding layer on the corresponding input.

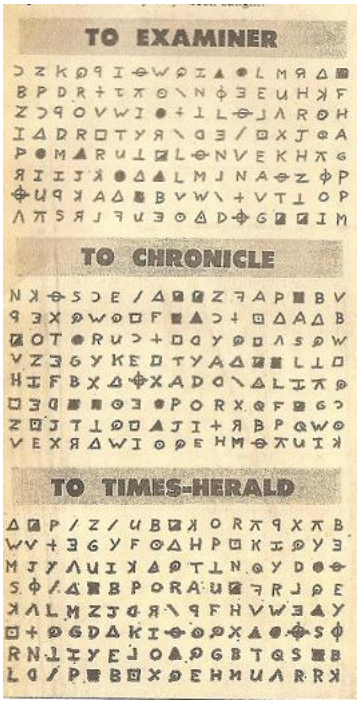


Figure 15: The original full version of the **Zodiac-408** cipher published by the San Francisco Examiner on August 3rd 1969.

Figure 16: Page 1 of the **BnF-f01** cipher from 1500s.

Figure 17: Page 2 of the **BnF-f01** cipher from 1500s.