Decipherment as Regression

Solving Historical Substitution Ciphers by Learning Symbol Recurrence Relations

Nishant Kambhatla, Logan Born, Anoop Sarkar

May 2023

Findings of the Association for Computational Linguistics: EACL 2023

Presented by Morten Munk Andersen

Contents

1. Why this paper? 2
2. Methodology 4
   - 2.1 Recurrent Integer Sequences 5
   - 2.2 Generative Decipherment Model 7
3. Results 12
   - 3.1 Synthetic Cipher 13
   - 3.2 Z408 Cipher 14

1. Why this paper?

Relevancy

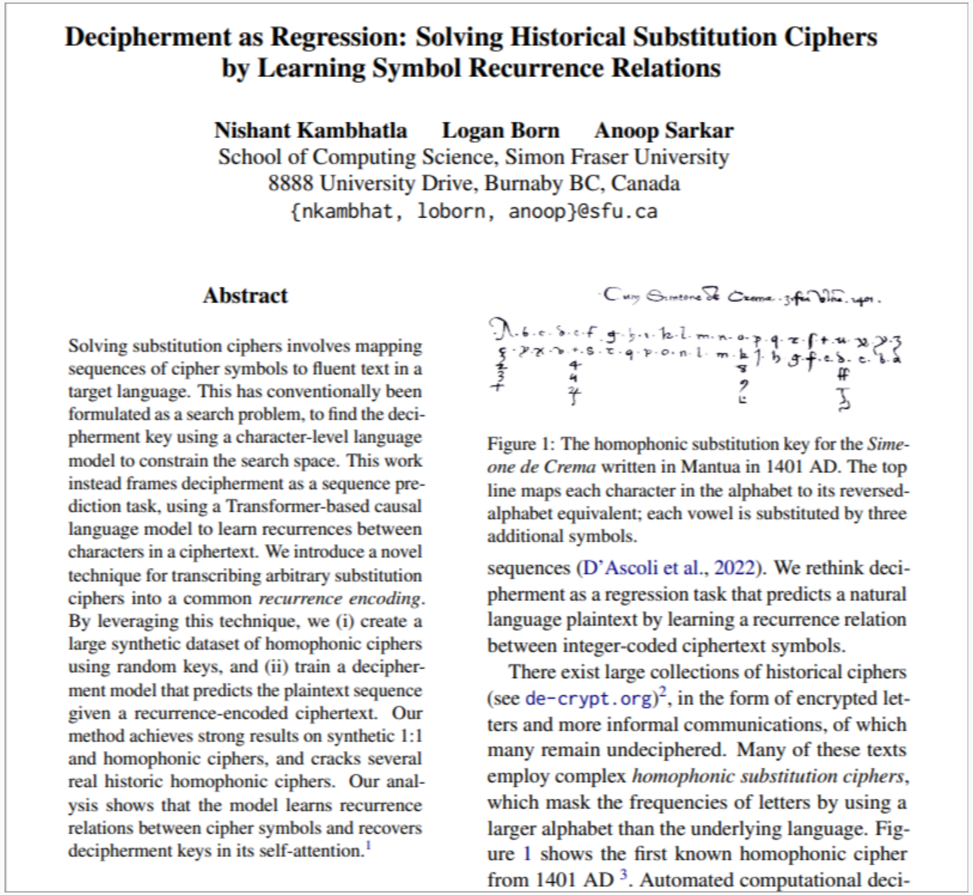
- Homophonic substitution ciphers

Ranking

- Core2023 Ranking: A

Recency

- May 2023



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

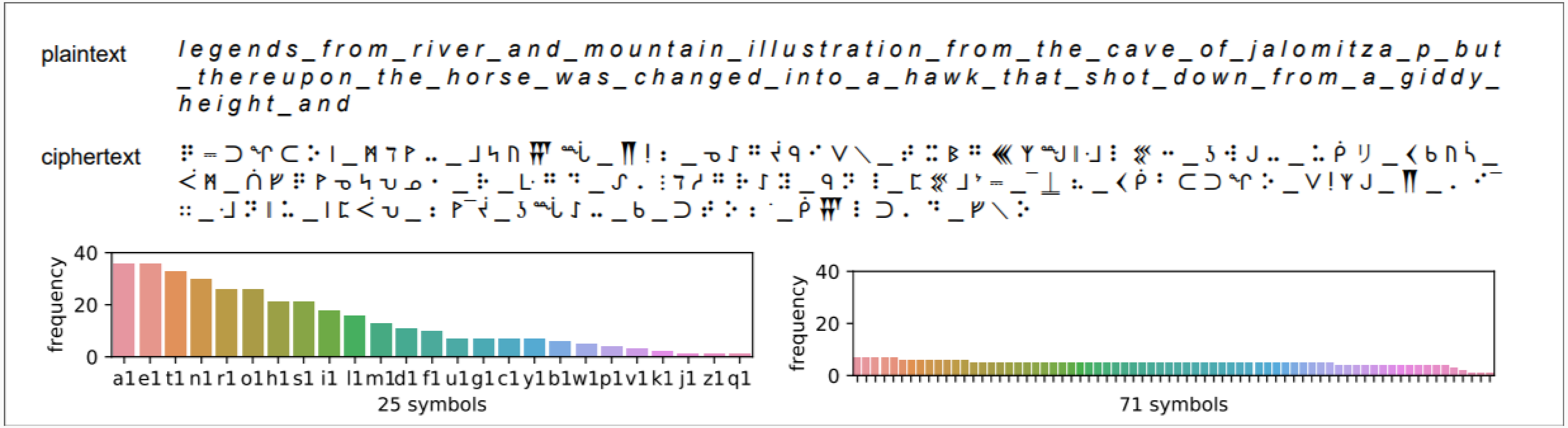
Solving substitution ciphers involves mapping sequences of cipher symbols to fluent text in a target language. This has conventionally been formulated as a search problem, to find the decipherment key using a character-level language model to constrain the search space. This work instead frames decipherment 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]

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.

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.

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 deci-

2. Methodology



Monoalphabetic (1:1)

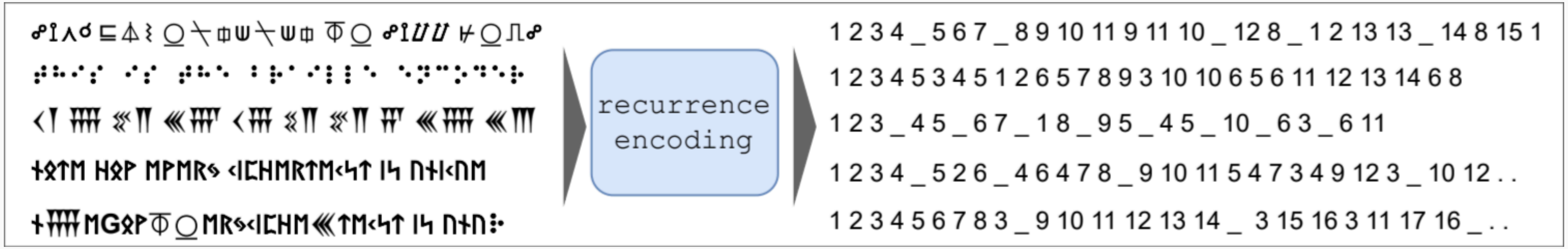
- Trivially solved with frequency analysis

Homophonic (1:>0)

- Harder to solve - frequencies can be hidden 😔
- More symbols = More mappings

Capturing first/repeated symbol occurences

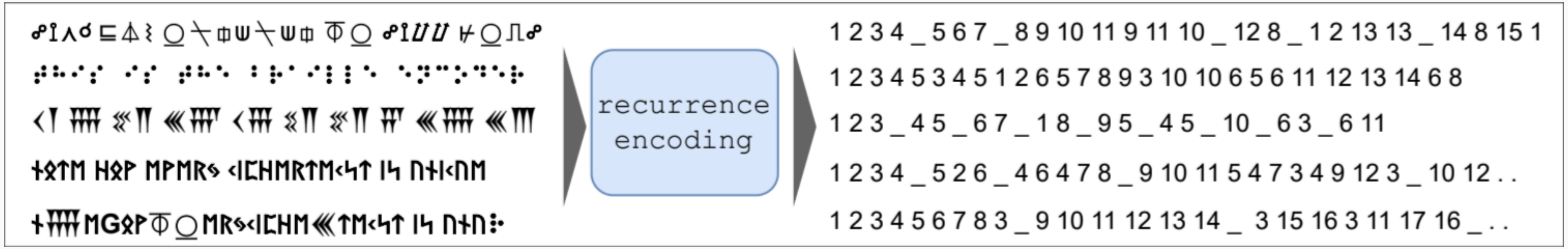
- Spaces denoted as **underscore**
- Unseen symbols denoted as **incremental integer**
- Recurring symbols denoted as represented **previous integer**
- Works for ciphers with different symbol sets



2.1 Recurrent Integer Sequences

Capturing first/repeated symbol occurences

- Spaces denoted as **underscore**
- Unseen symbols denoted as **incremental integer**
- Recurring symbols denoted as represented **previous integer**
- Works for ciphers with different symbol sets



The authors consider this a novel approach

2.2 Generative Decipherment Model

Remember: Ciphertext is now a Recurrent Integer Sequence

This makes every cipher comparable

Dataset made by authors

- 2 million unique homophonic substitution ciphers
- Including their corresponding plaintexts
- Uses Modern English

CausalLM

- Reads from left to right - can only look back
- Past words affect predicted words - (sort of like autocorrect)

CausalLM

- Reads from left to right - can only look back
- Past words affect predicted words - (sort of like autocorrect)

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

- $X^{l-1} \rightarrow \text{Cipher at layer previous to } l$
- $Y^{l-1} \rightarrow \text{Text at layer previous to } l$
- $\text{SelfAtn} \rightarrow \text{Captures positions related to previous symbols/letters}$
- $\text{Mask} \rightarrow \text{The attention mask used by SelfAtn}$
- $\text{FFN} \rightarrow \text{Result is fed to Feed-Forward Neural Network } X$

CausalLM

- Reads from left to right - can only look back
- Past words affect predicted words - (sort of like autocorrect)

$$[X^l, Y^l] = \mathrm{FFN} \circ \mathrm{SelfAtn}\big([X^{l-1}, Y^{l-1}], \mathrm{Mask}\big)$$

- $X^{l-1} \rightarrow$ Cipher at layer previous to l
- $Y^{l-1} \rightarrow$ Text at layer previous to l
- SelfAtn $\rightarrow$ Captures positions related to previous symbols/letters
- Mask $\rightarrow$ The attention mask used by SelfAtn
- FFN $\rightarrow$ Result is fed to Feed-Forward Neural Network X

Above produces the representation at $[X^l, Y^l]$

Remember: CausalLM only looks back!

Loss function

$$L^{\mathrm{CLM}}(X,Y) = L^{\mathrm{SRC}} + L^{\mathrm{TGT}} = -\mathrm{logP}(X) - \mathrm{logP}(Y|X)$$

- $L^{\mathrm{SRC}} \rightarrow$ Source loss - error predicting cipher seq
- $L^{\mathrm{TGT}} \rightarrow$ Target loss - error predicting plaintext seq
- $-\mathrm{logP}(X) \rightarrow$ Probability of reproducing correct cipher symbols
- $-\mathrm{logP}(X|Y) \rightarrow$ Probability of predicting plaintext given cipher

Loss function

$$L^{\text{CLM}}(X, Y) = L^{\text{SRC}} + L^{\text{TGT}} = -\log\text{P}(X) - \log\text{P}(Y|X)$$

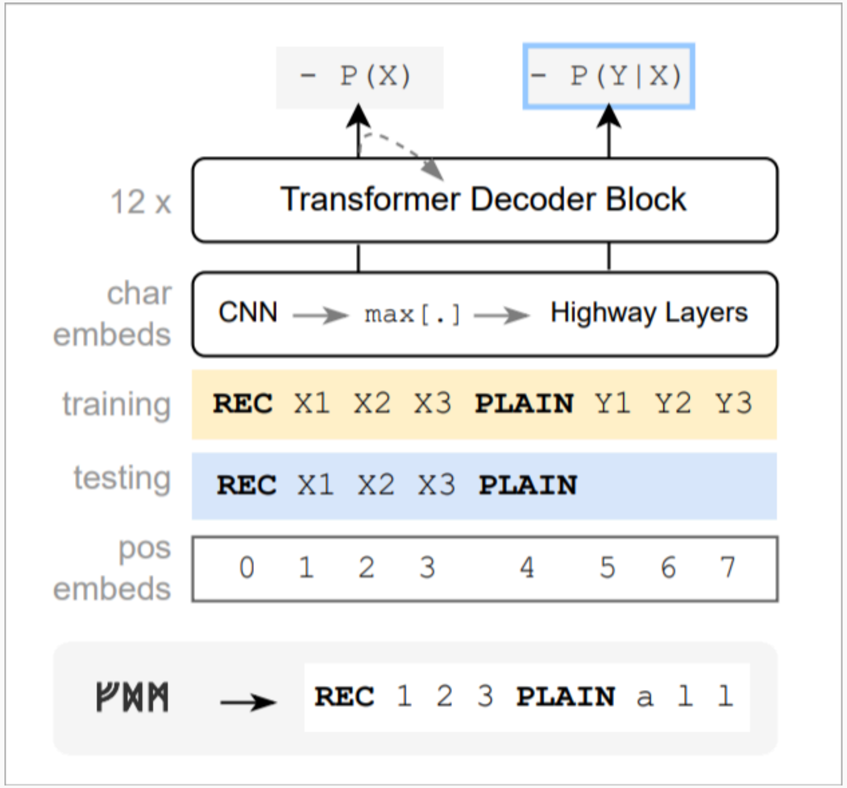
- $L^{\text{SRC}} \rightarrow$ Source loss - error predicting cipher seq
- $L^{\text{TGT}} \rightarrow$ Target loss - error predicting plaintext seq
- $-\log\text{P}(X) \rightarrow$ Probability of reproducing correct cipher symbols
- $-\log\text{P}(X|Y) \rightarrow$ Probability of predicting plaintext given cipher

Low probability = high loss, and vice versa

Probability can be seen as confidence

Why CausalLM?

- Predict the cipher symbols in a sequence
- Predict plaintext in the sequence
- Model learns the mappings



2.2 Generative Decipherment Model

Considered models

- Seq2seq
- Target-Only CausalLM
- PrefixLM

Why are they weaker?

- Only predicts plaintext
- Does not learn cipher symbol recurrence patterns

3. Results

How can we measure?

- Symbol Error Rate (SER)
- 0% = Perfect decipherment
- 100% Total gibberish

What can we observe?

- Between 400 - 700 chars
- Three key ranges
- CausalLM outperforms the others 🥳

#keys	**Model**	Max Len. **400**	**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**

How can we measure?

- Symbol Error Rate (SER)
- 0% = Perfect decipherment
- 100% Total gibberish

What can we observe?

- Between 400 - 700 chars
- Three key ranges
- CausalLM outperforms the others 🥳

#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**

Fun observation: seq2seq does not even converge at longer ciphers 🤣

3.2 Z408 Cipher

But what about real ciphers?

- Z408 = 408 characters
- 54 symbols
- From the 1960′s
- No spaces ❗

https://zodiackiller.fandom.com/wiki/408-cipher

3.2 Z408 Cipher

Hill-climbing

- Keep the best candidates

Beam search

- Keep N best candidates

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

Hill-climbing

- Keep the best candidates

Beam search

- Keep N best candidates

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

CausalLM 🎉

- Faster (it does not search)
- Better (even with smaller beam)