

Neural Decipherment via Minimum-Cost Flow: from Ugaritic to Linear B

Jiaming Luo
CSAIL, MIT

j_luo@csail.mit.edu

Yuan Cao
Google Brain

yuancao@google.com

Regina Barzilay
CSAIL, MIT

regina@csail.mit.edu

Abstract

In this paper we propose a novel neural approach for automatic decipherment of lost languages. To compensate for the lack of strong supervision signal, our model design is informed by patterns in language change documented in historical linguistics. The model utilizes an expressive sequence-to-sequence model to capture character-level correspondences between cognates. To effectively train the model in an unsupervised manner, we innovate the training procedure by formalizing it as a minimum-cost flow problem. When applied to the decipherment of Ugaritic, we achieve a 5.5% absolute improvement over state-of-the-art results. We also report the first automatic results in deciphering Linear B, a syllabic language related to ancient Greek, where our model correctly translates 67.3% of cognates.¹

1 Introduction

Decipherment is an ultimate low-resource challenge for both humans and machines. The lack of parallel data and scarce quantities of ancient text complicate the adoption of neural methods that dominate modern machine translation. Even for human experts this translation scenario proved to be onerous: a typical decipherment spans over decades and requires encyclopedic domain knowledge, prohibitive manual effort and sheer luck (Robinson, 2002). Moreover, techniques applied for the decipherment of one lost language are rarely reusable for another language. As a result, every significant human decipherment is considered to be one of a kind, “the rarest category of achievement” (Pope, 1975).

Prior work has demonstrated the feasibility of automatic decipherment. Snyder et al. (2010)

¹Code and all datasets are hosted in <https://github.com/j-luo93/NeuroDecipher>.

translated the ancient Semitic language Ugaritic into Hebrew. Since both languages are derived from the same proto-Semitic origin, the translation involved matching their alphabets at the character level and mapping cognates at the word level. The effectiveness of their approach stemmed from its ability to incorporate expansive linguistic knowledge, including expected morphological correspondences, the nature of alphabet-level alignment, etc. As with human decipherment, this approach is highly customized for a given language pair and does not generalize to other lost languages.

In this paper, we introduce a neural decipherment algorithm that delivers strong performances across several languages with distinct linguistic characteristics. As in prior work, our input consists of text in a lost language and a non-parallel corpus in a known related language. The model is evaluated on the accuracy of aligning words from the lost language to their counterparts in the known language.

To maintain the language-independent nature of the approach, we want to build the model around the most basic decipherment principles applicable across multiple languages. These principles are informed by known patterns in language change extensively documented in historical linguistics (Campbell, 2013). At the character level, we know that characters that originate from the same proto-language have similar distributional profiles with respect to their occurrences. Another important constraint at the character level is that cognate alignment is monotonic since character reorderings within cognate pairs are rare. At the vocabulary level, we want to enforce skewed mapping at the word level assuming roughly one-to-one correspondence. Finally, we want to ensure that the resulting vocabulary mapping covers a significant portion of the lost language vocabu-

lary and can also account for the presence of words which are not cognates.

Our model captures both character-level and word-level constraints in a single generative framework wherein vocabulary level alignment is a latent variable. We model cognate generation process using a character-level sequence-to-sequence model which is guided towards monotonic rewriting via regularization. Distributional similarity at the character level is achieved via universal character embeddings. We enforce constraints on the vocabulary mapping via minimum-cost flow formulation that controls structural sparsity and coverage on the global cognate assignment. The two components of the model – sequence-to-sequence character alignment and flow constraints – are trained jointly using an EM-style procedure.

We evaluate our algorithm on two lost languages – Ugaritic and Linear B. In the case of Ugaritic, we demonstrate improved performance of cognate identification, yielding 5.5% absolute improvement over previously published results (Snyder et al., 2010). This is achieved without assuming access to the morphological information in the known language.

To demonstrate the applicability of our model to other linguistic families, we also consider decipherment of Linear B, an ancient script dating back to 1450BC. Linear B exhibits a number of significant differences from Ugaritic, most noted among them its syllabic writing system. It has not been previously deciphered by automatic means. We were able to correctly translate 67.3% of Linear B cognates into their Greek equivalents in the decipherment scenario. Finally, we demonstrate that the model achieves superior performance on cognate datasets used in previous work (Berg-Kirkpatrick and Klein, 2013).

2 Related Work

Decoding of Ciphred Texts Early work on decipherment was primarily focused on man-made ciphers, such as substitution ciphers. Most of these approaches are based on EM algorithms which are further adjusted for target decipherment scenarios. These adjustments are informed by assumptions about ciphers used to produce the data (Knight and Yamada, 1999; Knight et al., 2006; Ravi and Knight, 2011; Pourdamghani and Knight, 2017). Besides the commonly used EM algorithm, (Nuhn

et al., 2013; Hauer et al., 2014; Kambhatla et al., 2018) also tackles substitution decipherment and formulate this problem as a heuristic search procedure, with guidance provided by an external language model (LM) for candidate rescoring. So far, techniques developed for man-made ciphers have not been shown successful in deciphering archaeological data. This can be attributed to the inherent complexity associated with processes behind language evolution of related languages.

Nonparallel Machine Translation Advancements in distributed representations kindled exciting developments in this field, including translations at both the lexical and the sentence level. Lexical translation is primarily formulated as alignment of monolingual embedding spaces into a crosslingual representation using adversarial training (Conneau et al., 2017), VAE (Dou et al., 2018), CCA (Haghighi et al., 2008; Faruqui and Dyer, 2014) or mutual information (Mukherjee et al., 2018). The constructed monolingual embedding spaces are usually of high quality due to the large amount of monolingual data available. The improved quality of distributed representations has similarly strong impact on non-parallel translation systems that operate at the sentence level (Pourdamghani and Knight, 2017). In that case, access to a powerful language model can partially compensate for the lack of explicit parallel supervision. Unfortunately, these methods cannot be applied to ancient texts due to the scarcity of available data.

Decoding of Ancient Texts (Snyder et al., 2010) were the first to demonstrate the feasibility of automatic decipherment of a dead language using non-parallel data. The success of their approach can be attributed to cleverly designed Bayesian model that structurally incorporated powerful linguistic constraints. This includes customized priors for alphabet matching, incorporation of morphological structure, etc. (Berg-Kirkpatrick and Klein, 2011) proposed an alternative decipherment approach based on a relatively simple model paired with sophisticated inference algorithm. While their model performed well in a noise-free scenario when matching vocabularies only contain cognates, it has not been shown successful in a full decipherment scenario. Our approach outperforms these models in both scenarios. Moreover, we have demonstrated that the

same architecture deciphers two distinct ancient languages Ugaritic and Linear B. The latter result is particularly important given that Linear B is a syllabic language.

3 Approach

The main challenge of the decipherment task is the lack of strong supervision signal that guides standard machine translation algorithms. Therefore, the proposed architecture has to effectively utilize known patterns in language change to guide the decipherment process. These properties are summarized below:

1. *Distributional Similarity of Matching Characters*: Since matching characters appear in similar places in corresponding cognates, their contexts should match.
2. *Monotonic Character Mapping within Cognates*: Matching cognates rarely exhibit character reordering, therefore their alignment should be order preserving.
3. *Structural Sparsity of Cognate Mapping*: It is well-documented in historical linguistics that cognate matches are mostly one-to-one, since both words are derived from the same proto-origin.
4. *Significant Cognate Overlap Within Related Languages*: We expect that the derived vocabulary mapping will have sufficient coverage for lost language cognates.

3.1 Generative framework

We encapsulate these basic decipherment principles into a single generative framework. Specifically, we introduce a latent variable $\mathcal{F} = \{f_{i,j}\}$ that represents the word-level alignment between the words in the lost language $\mathcal{X} = \{x_i\}$ and those in the known language $\mathcal{Y} = \{y_j\}$. More formally, we derive the joint probability

$$\begin{aligned} \Pr(\mathcal{X}, \mathcal{Y}) &= \sum_{\mathcal{F} \in \mathbb{F}} \Pr(\mathcal{F}) \Pr(\mathcal{X}|\mathcal{F}) \Pr(\mathcal{Y}|\mathcal{F}, \mathcal{X}) \\ &\propto \sum_{\mathcal{F} \in \mathbb{F}} \Pr(\mathcal{Y}|\mathcal{X}, \mathcal{F}) \\ &= \sum_{\mathcal{F} \in \mathbb{F}} \prod_{y_j \in \mathcal{Y}} \Pr(y_j|\mathcal{X}, \mathcal{F}), \end{aligned} \quad (1)$$

by assuming a uniform prior on both $\Pr(\mathcal{F})$ and $\Pr(\mathcal{X}|\mathcal{F})$, and i.i.d. for every $y_j \in \mathcal{Y}$. We use \mathbb{F} to

describe the set of valid values for the latent variable \mathcal{F} , subject to the global constraints as stated in Property 3 and 4. More specifically, we utilize a minimum-cost flow setup to enforce these properties.

The probability distribution $\Pr(y_j|\mathcal{X}, \mathcal{F})$ is further defined as

$$\Pr(y_j|\mathcal{X}, \mathcal{F}) = \sum_{x_i \in \mathcal{X}} f_{i,j} \cdot \Pr_{\theta}(y_j|x_i), \quad (2)$$

where the conditional probability $\Pr_{\theta}(y_j|x_i)$ is modeled by a character-based neural network parameterized by θ , which incorporates the character-level constraints as stated in Property 1 and 2.

Directly optimizing Equation (1) is infeasible since it contains a summation over all valid flows. To bypass this issue, we adopt an EM-style iterative training regime. Specifically, the training process involves two interleaving steps. First, given the value of the flow \mathcal{F} , the neural model is trained to optimize the likelihood function $\prod_{y_j \in \mathcal{Y}} \Pr(y_j|\mathcal{X}, \mathcal{F})$. Next, the flow is updated by solving a minimum-cost flow problem given the trained neural model. A detailed discussion of the training process is presented in Section 4.

We now proceed to provide details on both the neural model and the minimum-flow setup.

3.2 Neural decipherment model

We use a character-based sequence-to-sequence (seq2seq) model to incorporate the local constraints (Figure 1). Specifically, we integrate Property 1 by using a shared universal character embedding space and a residual connection. Furthermore, the property of monotonic rewriting is realized by a regularization term based on edit distance. We detail each component in the following paragraphs.

Universal character embedding We directly require that character embeddings of the two languages reside in the same space. Specifically, we assume that any character embedding in a given language is a linear combination of universal embeddings. More formally, we use a universal embedding matrix $U \in M^{n_u \times d}$, a lost language character weight matrix $W_x \in M^{n_x \times n_u}$ and a known language character weight matrix $W_y \in M^{n_y \times n_u}$. We use n_u to denote the size of the universal character inventory, and n_x, n_y the number of unique

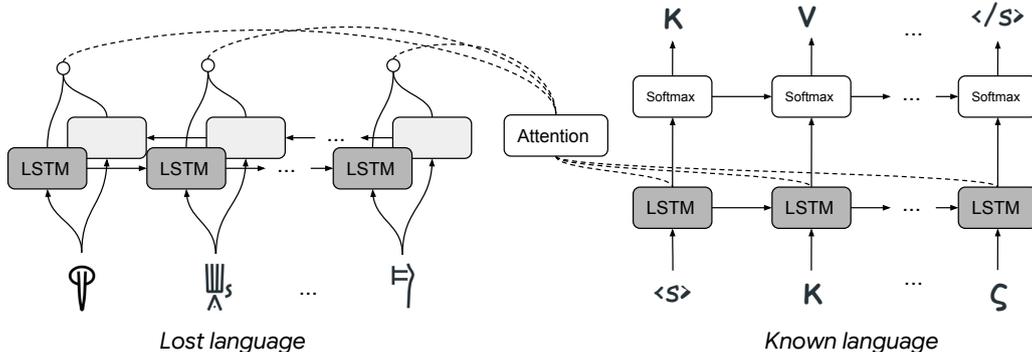


Figure 1: Architecture of our proposed model. For simplicity, we omit lines for residual connections linking weighted sum of input embeddings and softmax. Inputs to the encoder and decoder are the lost and known languages respectively. See Sec. 3.2 for details.

characters in the lost and the known languages, respectively. Embedding matrices for both languages are computed by

$$E_x = W_x U,$$

$$E_y = W_y U.$$

This formulation reflects the principle underlying crosslingual embeddings such as MUSE (Conneau et al., 2017). Along a similar line, previous work has demonstrated the effectiveness of using universal word embeddings, in the context of low-resource neural machine translation (Gu et al., 2018).

Residual connection Character alignment is mostly local in nature, but this fact is not reflected by how the next character is predicted by the model. Specifically, the prediction is made based on the context vector \tilde{h} , which is a nonlinear function of the hidden states of the encoder and the decoder. As a result, \tilde{h} captures a much wider context due to the nature of a recurrent neural network.

To address this issue and directly improve the quality of character alignment, we add a residual connection from the encoder embedding layer to the decoder projection layer. Specifically, letting α be the predicted attention weights, we compute

$$c = \sum_i \alpha_i E_x(i),$$

$$\hat{h} = c \oplus \tilde{h}, \quad (3)$$

where $E_x(i)$ is the encoder character embedding at position i , and c is the weighted character embedding. \hat{h} is subsequently used to predict the next character. A similar strategy has also been adopted

		ϕ	Α	ς	(Linear B)
(Greek)	K	✓			
	V		✗		
	W		✓		
	σ			✓	
	ο			✓	
	ς			✗	

Figure 2: An example of alignment between a Linear B word and Greek word. ✓ and ✗ denote correct and wrong alignment positions respectively. The misalignment between Α and ν incurs a deletion error; ϕ and ζ incurs an insertion error.

by Nguyen and Chiang (2018) to refine the quality of lexical translations in NMT.

Monotonic alignment regularization We design a regularization term that guides the model towards monotonic rewriting. Specifically, we penalize the model whenever insertions or deletions occur. More concretely, for each word in the lost language x_i , we first compute the alignment probability $\Pr(a_i^t | x_i)$ over the input sequence at decoder time step t , predicted by the attention mechanism. Then we compute the expected alignment position as

$$p_i^t = \sum_k k \cdot \Pr(a_i^t = k | x_i),$$

where k is any potential aligned position. The regularization term is subsequently defined as

$$\Omega_1(\{p_i^t\}) = \sum_t (p_i^t - p_i^{t-1} - 1)^2. \quad (4)$$

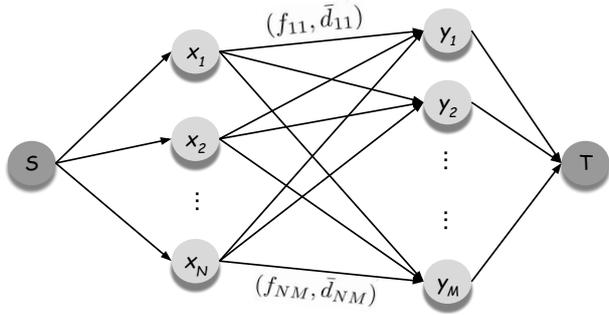


Figure 3: Minimum-cost flow. S, T stands for source and sink respectively; x_i, y_j are the i^{th} and j^{th} word in \mathcal{X} and \mathcal{Y} . Each edge is associated with a flow f_{ij} and cost \bar{d}_{ij} . See Sec. 3.3 for details.

Note that no loss is incurred when the current alignment position immediately follows the previous position, namely $p_i^t = p_i^{t-1} + 1$. Furthermore, we use a quadratic loss function to discourage expensive multi-character insertions and deletions.

For Linear B, we modify this regularization term to accommodate the fact that it is a syllabic language and usually one linear B script corresponds to two Greek letters. Particularly, we use the following regularization term for Linear B

$$\Omega_2(\{p_i^t\}) = \sum_{t=1} (p_i^t - p_i^{t-2} - 1)^2. \quad (5)$$

Figure 2 illustrates one alignment matrix from Linear B to Greek. In this example, the Linear B character 𐀀 is supposed to be aligned with Greek characters ν and ω but only got assigned to ω , hence incurring a deletion error; 𐀁 is supposed to be only aligned to σ and o , but assigned an extra alignment to ζ , incurring an insertion error.

3.3 Minimum-cost flow

The latent variable \mathcal{F} captures the global constraints as stated in Property 3 and 4. Specifically, \mathcal{F} should identify a reasonable number of cognate pairs between the two languages, while meeting the requirement that word-level alignments are one-to-one. To this end, we cast the task of identifying cognate pairs as a minimum-cost flow problem (Figure 3). More concretely, we have three sets of edges in the flow setup:

- $f_{s,i}$: edges from the source node to the word x_i in the lost language,
- $f_{j,t}$: edges from the word y_j in the known language to the sink node,

- $f_{i,j}$: edges from x_i to y_j .

Each edge has a capacity of 1, effectively enforcing the one-to-one constraint. Only the edges $f_{i,j}$ have associated costs. We define this cost as the expected distance between x_i and y_j :

$$\bar{d}_{i,j} = \mathbb{E}_{y \sim \text{Pr}(y|x_i)} d(y, y_j), \quad (6)$$

where $d(\cdot, \cdot)$ is the edit distance function, and $\text{Pr}(y|x_i)$ is given by the neural decipherment model. We use a sampling procedure proposed by Shen et al. (2016) to compute this expected distance. To provide a reasonable coverage of the cognate pairs, we further specify the demand constraint $\sum_j f_{j,t} = D$ with a given hyperparameter D .

We note that the edit distance cost plays an essential role of complementing the neural model. Specifically, neural seq2seq models are notoriously inadequate at capturing insertions and deletions, contributing to many issues of overgeneration or undergeneration in NMT (Tu et al., 2016). These problems are only accentuated due to a lack of supervision. Using edit distance in the flow setup helps alleviate this issue, since a misstep of insertion or deletion by the neural model will still generate a string that resembles the ground truth in terms of edit distance. In other words, the edit distance based flow can still recover from the mistakes the neural model makes.

4 Training

We note that with weak supervision, a powerful neural model can produce linguistically degenerate solutions. To prevent the neural model from getting stuck at an unreasonable local minimum, we make three modifications detailed in the following paragraphs. The entire training procedure is illustrated in Alg 1.

Flow decay The flow solver returns sparse values – the flow values for the edges are mostly zero. It is likely that this will discard many true cognate pairs, and the neural model trained on these sparse values can be easily misled and get stuck at some suboptimal local minimum.

To alleviate this issue, we apply an exponential decay to the flow values, and compute an interpolation between the new flow result and the previous one. Specifically, we update the flow at iteration τ as

$$f_{i,j}^{(\tau)} = \gamma \cdot f_{i,j}^{(\tau-1)} + (1 - \gamma) \cdot \tilde{f}_{i,j}^{(\tau)}, \forall i, j, \quad (7)$$

Algorithm 1 Iterative training

Require:

\mathcal{X}, \mathcal{Y} : vocabularies,
 T : number of iterations,
 N : number of cognate pairs to identify.

```
1:  $f_{i,j}^{(0)} \leftarrow \frac{N}{|\mathcal{X}| \cdot |\mathcal{Y}|}$   $\triangleright$  Initialize
2: for  $\tau \leftarrow 1$  to  $T$  do
3:    $\theta^{(\tau)} \leftarrow \text{MLE-TRAIN}(f_{i,j}^{(\tau-1)})$ 
4:    $\tilde{d}_{i,j}^{(\tau)} \leftarrow \text{EDIT-DIST}(x_i, y_j, \theta^{(\tau)})$ 
5:    $\tilde{f}_{i,j}^{(\tau)} \leftarrow \text{MIN-COST-FLOW}(\tilde{d}_{i,j}^{(\tau)})$ 
6:    $f_{i,j}^{(\tau)} \leftarrow \gamma \cdot f_{i,j}^{(\tau-1)} + (1 - \gamma) \cdot \tilde{f}_{i,j}^{(\tau)}$ 
7:   RESET( $\theta^{(\tau)}$ )
8: return  $f_{i,j}^{(T)}$ 

9: function  $\text{MLE-TRAIN}(f_{i,j}^{(\tau)})$ 
10:    $\theta^{(\tau)} \leftarrow \arg \max_{\theta} \prod_{y_j \in \mathcal{Y}} \Pr_{\theta}(y_j | \mathcal{X}, \mathcal{F})$ 
11:   return  $\theta^{(\tau)}$ 
```

where $\tilde{f}_{i,j}^{(\tau)}$ is the raw output given by the flow solver, and γ is a hyperparameter.

Norm control Recall that the residual connection combines a weighted character embedding c , and a context vector \tilde{h} (Equation (3)). We observe that during training, \tilde{h} has a much bigger norm than c , essentially defeating the purpose of improving character alignment by using a residual connection. To address this issue, we rescale \tilde{h} so that the norm of \tilde{h} does not exceed a certain percentage of the norm of c . More formally, given a ratio $r < 1.0$, we compute the residual output as

$$\hat{h} = c \oplus (g \cdot \tilde{h})$$
$$g = \min(r * \frac{\|c\|_2}{\|\tilde{h}\|_2}, 1.0)$$

Periodic reset We re-initialize the parameters of the neural model and reset the state of the optimizer after each iteration. Empirically, we found that our neural network can easily converge to a suboptimal local minimum given a poor global word-level alignment. Resetting the model parameters periodically helps with limiting the negative effect caused by such alignments.

5 Experiments

Datasets We evaluate our system on the following datasets:

- **UGARITIC:** Decipherment from Ugaritic to Hebrew. Ugaritic is an ancient Semitic language closely related to Hebrew, which was used for the decipherment of Ugaritic. This dataset has been previously used for decipherment by Snyder et al. (2010).

- **Linear B:** Decipherment from Linear B to Greek. Linear B is a syllabic writing system used to write Mycenaean Greek dating back to around 1450BC. Decipherment of a syllabic language like Linear B is significantly harder, since it employs a much bigger inventory of symbols (70 in our corpus), and the symbols that have the same consonant or vowel look nothing alike².

We extracted pairs of Linear B scripts (i.e., words) and Greek pronunciations from a compiled list of Linear B lexicon³. We process the data by removing some uncertain translations, eventually retaining 919 pairs in total. The linear B scripts are kept as it is, and we remove all diacritics in the Greek data.

We also consider a subset of the Greek data to simulate an actual historical event where many linear B syllabograms were deciphered by being compared with Greek location names. On the Greek side, we retain 455 proper nouns such as locations, names of Gods or Goddesses, and personal names. The entire vocabulary of the Linear B side is kept as it is. This results in a dataset with roughly 50% unpaired words on the Linear B side. We call this subset Linear B/names.

To the best of our knowledge, our experiment is the first attempt of deciphering Linear B automatically.

- **ROMANCE:** Cognate detection between three Romance languages. It contains phonetic transcriptions of cognates in Italian, Spanish and Portuguese. This dataset has been used by Hall and Klein (2010) and Berg-Kirkpatrick and Klein (2011).

Data statistics are summarized in Table 1.

²For instance, \oplus , \otimes and \oplus encode “ka”, “ke” and “te”, respectively.

³<https://archive.org/details/LinearBlexicon/page/n5>

Dataset	#Cognates	#Tokens (lost/known)	#Symbols (lost/known)
UGARITIC	2214	7353/41263	30/23
Linear B	919	919/919	70/28
Linear B/names	455	919/455	70/28
ROMANCE	583	583/583	25/31/28 (Es/It/Pt)

Table 1: Statistics of datasets used in our experiments.

Systems We report numbers for the following systems:

- **Bayesian**: the Bayesian model by Snyder et al. (2010) that automatically deciphered Ugaritic to Hebrew
- **Matcher**: the system using combinatorial optimization, proposed by Berg-Kirkpatrick and Klein (2011).
- **NeuroCipher**: our proposed model.

We directly quote numbers from their papers for the UGARITIC and ROMANCE datasets. To facilitate direct comparison, we follow the same data processing procedure as documented in the literature.

Training details Our neural model uses a bidirectional-LSTM as the encoder and a single-layer LSTM as the decoder. The dimensionality of character embeddings and the hidden size of LSTM are set to 250 for all experiments. The size of the universal character inventory is 50 for all datasets except Linear B for which we use 100. The hyperparameter for alignment regularization is set to 0.5, and the ratio r to control the norm of the context vector is set to 0.2. We use ADAM (Kingma and Ba, 2015) to optimize the neural model. To speed up the process of solving the minimum-cost flow problem, we sparsify the flow graph by only considering the top 5 candidates for every x_i . $\gamma = 0.9$ is used for the flow decay on all datasets except on UGARITIC for which we use $\gamma = 0.25$. We use the OR-Tools optimization toolkit⁴ as the flow solver.

We found it beneficial to train our model only on a randomly selected subset (10%) of the entire corpus with the same percentage of noncognates, and test it on the full dataset. It is common for the dataset UGARITIC to contain several cognates for the same Ugaritic word, and we found that

relaxing the capacity $f_{j,t}$ to 3 yields a better result. For Linear B, similar to the finding by (Berg-Kirkpatrick and Klein, 2013), random restarting and choosing the best model based on the objective produces substantial improvements. In scenarios where many unpaired cognates are present, we follow Haghghi et al. (2008) to gradually increase the number of cognate pairs to identify.

6 Results

UGARITIC We evaluate our system in two settings. First, we test the model under the noiseless condition where only cognates pairs are included during training. This is the setting adopted by Berg-Kirkpatrick and Klein (2011). Second, we conduct experiments in the more difficult and realistic scenario where there are unpaired words in both Ugaritic and Hebrew. This is the noisy setting considered by Snyder et al. (2010). As summarized by Table 2, our system outperforms existing methods by 3.1% under the noiseless condition, and 5.5% under the noisy condition.

We note that the significant improvement under the noisy condition is achieved without assuming access to any morphological information in Hebrew. In contrast, previous system Bayesian utilized an inventory of known morphemes and complete morphological segmentations in Hebrew during training. The significant gains in identifying cognate pairs suggest that our proposed model provide a strong and viable approach towards automatic decipherment.

System	Noiseless	Noisy
Matcher	90.4	-
Bayesian	-	60.4
NeuroCipher	93.5	65.9

Table 2: Cognate identification accuracy (%) for UGARITIC under noiseless and noisy conditions. The noiseless baseline result is quoted from (Berg-Kirkpatrick and Klein, 2011), and the noisy baseline result is quoted from (Snyder et al., 2010).

⁴<https://github.com/google/or-tools>

System	Linear B	Linear B/names
NeuroCipher	84.7	67.3

Table 3: Cognate identification accuracy (%) for LinearB under noiseless and noisy conditions.

Linear B To illustrate the applicability of our system to other linguistic families, we evaluate the model on Linear B and Linear B/names. Table 3 shows that our system reaches high accuracy at 84.7% in the noiseless LinearB corpus, and 67.3% accuracy in the more challenging and realistic LinearB-names dataset.

We note that our system is able to achieve a reasonable level of accuracy with minimal change to the system. The only significant modification is the usage of a slightly different alignment regularization term (Equation (5)). We also note that this language pair is not directly applicable to both of the previous systems *Bayesian* and *Matcher*. The flexibility of the neural decipherment model is one of the major advantages of our approach.

ROMANCE Finally, we report results for ROMANCE (Hall and Klein, 2010) in Table 4, as further verification of the efficacy of our system. We include the average cognate detection accuracy across all language pairs as well as the accuracies for individual pairs. Note that in this experiment the dataset does not contain unpaired words. Table 4 shows that our system improves the overall accuracy by 1.5%, mostly contributed by $Es \Rightarrow It$ and $It \Rightarrow Pt$.⁵

System	EsIt	EsPt	ItPt	Avg
<i>Matcher</i>	88.9	95.6	85.7	90.1
NeuroCipher	92.3	95.0	87.3	91.6

Table 4: Cognate identification accuracy (%) for ROMANCE. Avg means the average accuracy across all six language pairs. EsIt, EsPt, ItPt are average accuracy for each language pair respectively (Es=Spanish, It=Italian, Pt=Portuguese). Results for *Matcher* are quoted from (Berg-Kirkpatrick and Klein, 2011).

Ablation study Finally, we investigate contribution of various components of the model architecture to the decipherment performance. Specifically, we look at the design choices directly in-

⁵We nevertheless observed a slight drop for $Es \Rightarrow Pt$. However, for this language pair, the absolute accuracy is already very high ($\geq 95\%$). We therefore suspect that performance on this language pair is close to saturation.

System	UGARITIC
NeuroCipher	65.9
-monotonic	0.0
-residual	0.0
-flow	8.6

Table 5: Results for the noisy setting of UGARITIC. -monotonic and -residual remove the monotonic alignment regularization and the residual connection, and -flow does not use flow or iterative training.

formed by patterns in language change: In all the above cases, the reduced decipherment model fails. The first two cases reach 0% accuracy, and the third one barely reaches 10%. This illustrates the utmost importance of injecting prior linguistic knowledge into the design of modeling and training, for the success of decipherment.

7 Conclusions

We proposed a novel neural decipherment approach. We design the model and training procedure following fundamental principles of decipherment from historical linguistics, which effectively guide the decipherment process without supervision signal. We use a neural sequence-to-sequence model to capture character-level cognate generation process, for which the training procedure is formulated as flow to impose vocabulary-level structural sparsity. We evaluate our approach on two lost languages, Ugaritic and Linear B, from different linguistic families, and observed substantially high accuracy in cognate identification. Our approach also demonstrated significant improvement over existing work on Romance languages.

Acknowledgments

This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via contract # FA8650-17-C-9116. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein. The authors are also grateful for the support of MIT Quest for Intelligence program.

References

- Taylor Berg-Kirkpatrick and Dan Klein. 2011. Simple effective decipherment via combinatorial optimization. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 313–321. 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. Association for Computational Linguistics.
- Lyle Campbell. 2013. *Historical Linguistics: An Introduction*. Edinburgh University Press.
- Alexis Conneau, Guillaume Lample, Marc’Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2017. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*.
- Zi-Yi Dou, Zhi-Hao Zhou, and Shujian Huang. 2018. Unsupervised bilingual lexicon induction via latent variable models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 621–626. Association for Computational Linguistics.
- Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 462–471. Association for Computational Linguistics.
- Jiatao Gu, Hany Hassan, Jacob Devlin, and Victor OK Li. 2018. Universal neural machine translation for extremely low resource languages. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 344–354.
- Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *Proceedings of ACL-08: HLT*, pages 771–779. Association for Computational Linguistics.
- David Hall and Dan Klein. 2010. Finding cognate groups using phylogenies. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1030–1039. Association for Computational Linguistics.
- 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 City University and 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. Association for Computational Linguistics.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.
- 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. Association for Computational Linguistics.
- Kevin Knight and Kenji Yamada. 1999. A computational approach to deciphering unknown scripts. In *Unsupervised Learning in Natural Language Processing*.
- Tanmoy Mukherjee, Makoto Yamada, and Timothy Hospedales. 2018. Learning unsupervised word translations without adversaries. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 627–632.
- Toan Nguyen and David Chiang. 2018. Improving lexical choice in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 334–343.
- 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. Association for Computational Linguistics.
- Maurice Pope. 1975. *The Story of Decipherment: From Egyptian Hieroglyphic to Linear B*. Thames & Hudson.
- Nima Pourdamghani and Kevin Knight. 2017. Deciphering related languages. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2513–2518.
- 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-Volume 1*, pages 12–21. Association for Computational Linguistics.
- Andrew Robinson. 2002. *Lost languages: the enigma of the world’s undeciphered scripts*.
- Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum

risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1683–1692.

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. Association for Computational Linguistics.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 76–85.