Deep Transfer Learning for Language Generation from Limited Corpora

Ted Moskovitz
Advanced Topics in Deep Learning
Final Project, Fall 2017

Abstract

The limited size of surviving corpora is a major obstacle in the deciperhment of ancient writing systems. Fortunately, many unknown scripts either encode a known language or are closely related to one. The approach presented in this paper leverages this fact, demonstrating that a character-level recurrent neural network (RNN) can be pre-trained on a large corpus of a related language before being trained on a smaller corpus from a target language without a significant drop in language generation performance. The success of this approach is an example of the computational power and flexibility of such neural network models, and may have broader applications beyond decipherment.

1 Introduction

Modern computing is being increasingly applied to disciplines not traditionally associated with quantitative analysis. One relatively untested area of application, however, is that of historical epigraphy: the reading and decipherment of ancient scripts. Until recently, computers have not been considered capable of aiding in such problems, lacking a synthesis of "logic and intuition" [Robinson, 2002]. However, recent advances in machine learning and computational statistics have begun to change that. In his book *The Story of Decipherment: From Egyptian Hieroglyphs to Maya Script*, Maurice Pope identifies three key prerequisites for decipherment by analyzing the processes that have led to success in the past. They are: (1) confidence that the problem is solvable, (2) isolation of a limited target, and (3) the ascertainment of the rules of the script [Pope, 1975].

While the first two conditions are certainly important, computational techniques can most easily be applied to the third: determining the rules of the script. This problem is fundamentally different from the more familiar problem of translation, as the primary goal is to make the script readable—-it is more akin to text-to-speech conversion than machine translation [Yamada and Knight, 1999]. Additionally, decipherment problems can come in several forms. In some cases, the script in question uses a known writing system, but the language is unknown. Such was the case with Phoenician [Pope, 1975]. However, most undeciphered scripts today fall into the categories of either unknown writing systems and known

languages, or unknown writing systems and unknown languages. The model for decipherment presented in this paper addresses just this context, in which decipherers are left with an unknown script that encodes a (possibly) known language.

Perhaps the primary obstacle standing in the way of undeciphered writing systems is the limited size of their extant corpora. The goal of this project is to learn the inter-character relationships in an unknown script, and use that understanding (encoded in the parameters of a machine learning model) to produce more text in that writing system. The hope is that this artificially generated text can sufficiently capture the dynamics of the unknown writing system such that it can be used by linguists to increase their understanding of the script itself.

Deep learning approaches typically require a large volume of data on which to train. This poses a problem, as the limited size of extant corpora is the very issue at hand. Fortunately, geographical and historical knowledge of undeciphered scripts often provides clues as to which other languages they are related. Related writing systems often have similar character dynamics, and the novel approach presented here takes advantage of that fact by pre-training on the larger corpus of a related language/writing system before training on the target text.

As appropriately digitized corpora for ancient scripts are not readily available, this paper presents a proof-of-concept for this approach by testing on known writing systems.

2 Related Work

Previous computational approaches to decipherment have generally centered around the use of traditional machine learning techniques to generate mappings from the characters of an unknown writing system to characters in a known writing system. Previous work on Ugaritic, an ancient Semitic language found along what is now the Syrian coast, employed a hidden Markov model (HMM) to successfully match 22 out of 30 characters to their equivalents in modern day Hebrew [Knight et al., 2006], and more recent work has applied a non-parametric Bayesian model to the same problem, improving accuracy to 29 out of 30 characters successfully matched [Snyder et al., 2010]. Other approaches have used the Expectation Maximization (EM) algorithm to map characters in an unknown language to phonemes in the International Phonetic Alphabet (IPA) [Yamada and Knight, 1999]. However, these approaches have only been successfully applied to relatively basic decipherment problems with expansive corpora–Ugaritic is well-known as one of the most straightforward writing systems, having been primarily decoded just a year after its discovery in 1929 [Pope, 1975]. Comparatively no headway has been made on more challenging (still unsolved) scripts with a more limited number of surviving samples, such the Indus Script [Yadav et al., 2010], Linear A [Chadwick, 2014], and Rongorongo, the Easter Island script [Pozdniakov and Pozdniakov, 2007].

Most successful non-computational approaches, which have driven the decoding of virtually all previously unknown scripts, have taken advantage of the same idea that drives the method described here. With the exception of Ancient Egyptian, which had in the Rosetta Stone a parallel corpus from which to draw, writing systems like Linear B and Mayan hieroglyphics were solved largely through the insight that they either encode the same language as a known script [Chadwick, 2014], or are closely related to extant regional languages [Coe, 2012]. The method

proposed here take advantage of exactly that relationship, leveraging the structure of a known writing system encoding a known language as a means to gain insight into the workings of an unknown script.

The approach described in this paper does not seek to solve these writing systems directly, but instead facilitate their eventual decipherment–through either computational or traditional methods–by providing a method for increasing the size of extant corpora. In that sense, it is derived more directly from work on generative text models in deep learning [Sutskever et al., 2011], [Graves, 2013], [Mikolov and Zweig, 2012]. While recurrent neural networks (RNNs) are the standard for sequence processing in deep learning, recent work has also found success applying convolutional networks at a character-level to problems in natural language processing [Conneau et al., 2016]. However, this method is not as well-developed, and would likely be ill-suited to a generative task.

The method described here also takes advantage of the method of pre-training the models on a non-target corpus before tuning them on the unknown writing system itself. Pre-training has a long history in deep learning [Bengio et al., 2007], [Hinton et al., 2006], [Ranzato et al., 2008], [Erhan et al., 2010]. However, the nature of these methods differs from that which is applied here, as they employ unsupervised pre-training of each layer of the network prior to training on problem-relevant data. In fact, the term "pre-training," as it is typically used with regard to deep learning, would perhaps be a misnomer in this case. Instead, the method applied in this paper is to switch from training on a related dataset to the target dataset midway through training, as a means of compensating for the small size of the target. This is in some ways an attempt to circumvent the problem addressed by one-shot learning techniques, designed to train networks with minimal data [Santoro et al., 2016], [Vinyals et al., 2016].

3 Methods

3.1 Approach

My approach can be broken down into three steps: (1) showing that pre-training on a related corpus enables a deep recurrent network to effectively learn the underlying structure of the 'unknown' script, (2) demonstrating that the network is able to generate *novel* text sequences, and (3) that these novel character sequences could be useful for decipherment. Because there are currently no freely available comprehensive corpora for undeciphered scripts, I train the deep networks on known languages, in this case, English, Dutch, and an artificially generated numerical encoding of English (see Section 3.2). This carries the added benefit of making results verifiable–with a truly unknown writing system, it would not be possible to demonstrate this method as a proof-of-concept.

I operationalize the first step of my approach by comparing the performance of a network trained on a 'full' size corpus (about 470,000 characters) of the target writing system to the performance of a network pre-trained on a full-size corpus of a related writing system of the same size, and then trained on a 'small' corpus of the target writing system (around 7,800 characters). This size was chosen because that is roughly the size of the surviving corpus of the undeciphered Cretan script Linear A, a notoriously unsolved writing system known for the small size of its

surviving sample. The Shakespeare training corpus is cut into disjoint fragments of on average 40 characters to reflect the fragmented nature of the surviving samples of Linear A, which is broken up onto often unrelated clay tablets. I compare these results on both 'fragmented' corpora (where the target corpus is randomly shuffled; see 3.3) and 'non-fragmented' corpora, where the text is left un-shuffled. I also compare these results to those of a network trained only on the small target corpus to show that pre-training provides a significant boost in performance.

The second step is accomplished by generating sample text with the trained network(s), identifying the various n-grams present (at the word level), and noting what proportion of those were not observed in the training corpus. As even well-trained character-level language models are prone to the occasional spelling mistake, any novel unigrams present in the sample are removed prior to extracting higher level n-grams. This step removed on average only 1-2% of each 5,000 character sample, indicating that the networks were adapting well to the target corpus. N-grams are extracted for $n = 2$ to 15, thus spanning a range from basic phrasal constituents to complete sentences.

The third step is achieved by checking, for each level of n-gram, what proportion of the novel n-grams generated by the model are grammatical. Here, I define 'grammatical' as meaning they can be parsed using the Stanford constituency parser [Klein and Manning, 2003] to a non-fragment–that is, the parser is able to identify the n-gram as belonging to a constituency class (i.e. noun phrase, verb phrase, sentence, etc). This particular parser uses a probabilistic context-free grammar (PCFG) to achieve state-of-the-art accuracy in constituency parsing. Thus, when a constituent is successfully assigned, it is showing that the network has managed to generate a grammatical string of words not present in the training corpus. This is exactly the kind of production that would be useful for decipherment.

As indicated earlier, I undertook another experiment with a synthetic writing system consisting of a mapping of English characters to randomly selected numbers (details in Section 3.2). This was done to not only test the performance of the model on text that was written in the same language but encoded in a different form, but also to indirectly address another issue that arises in decipherment, character segmentation. Often when reading characters off of clay tablets, the writing is either faded, damaged, or both. To make matters more difficult, when the script is unknown, it can be difficult to tell where one character ends and another begins (see Figure 1). Because the encoded English could have overlapping digits, i.e. 'a' could map to '12' and 'b' could map to '23,' the character combination '123' is an ambiguous input to the network. The goal was to see how well the model was able to learn to disambiguate such writing given pre-training on English versus how well it learned to do so when trained only on the numerical encoding. Unfortunately, further analysis into the number of novel n-grams and which of those were grammatical was impeded by the ambiguity of the mapping itself–it proved difficult to create a reliable decoding function. However, further investigation is warranted.

Figure 1: An example of worn, damaged Linear A on a clay tablet min [2015].

3.2 Dataset Details

Three separate corpora were drawn from to train the networks. The base training corpus was a significant portion of *Harry Potter and the Sorceror's Stone*, by JK Rowling [Rowling, 1998], consisting of approximately 473,000 characters in modern English. The target training corpora (representing the 'unknown' writing systems) were a compilation of Shakespeare's plays (written during the fifteenth century) and the Dutch translation of Harriet Beecher Stowe's *Uncle Tom's Cabin* [Stowe, 1852][1], both obtained through Project Gutenberg [pro, 2017]. These corpora in their raw form consisted of approximately 1.1 million and 1.0 million characters, respectively, but were truncated to match the needs of the task. Dutch was chosen because it is, like English, a Germanic language, and therefore closely related. The comparison of Harry Potter and Shakespeare is also worth studying, as though they are written in the same language, they are separated by hundreds of years, containing very different syntax and vocabulary. There are many analogues in decipherment, such as the use of modern Greek to help decode Linear B [Chadwick, 2014].

For the HP - Encoded English combination, each alphabetical character (upper and lower case) was randomly mapped to a number between 1 and a parameter Λ I've termed the *inverse ambiguity degree*. This term is derived from the fact that a higher Λ corresponds to a lower chance for a collision between character mappings, i.e. when two characters are mapped to the same number. The probability of a

[1] Don't ask, it's the first Dutch book I found online

collision is simply

$$Pr(collision) = \frac{N}{\Lambda},$$

where N is the size of the alphabet (52 in this case, due to upper- and lower-casing). Thus, a higher Λ results in a less ambiguous text. The base text from which the encoded text was generated was the same Harry Potter book.

3.3 Pre-Processing

Another difficulty obstructing the decipherment of ancient writing systems with sparse corpora is the fractured nature of surviving samples. For example, the 7,800 surviving characters of Linear A are distributed among 1,427 samples, for an average of only 5.47 characters per tablet [Younger, 2000]. Slightly better is the corpus for Rongorongo, the Rapanui script, which consists of 26 texts containing approximately 15,000 characters, for an average of about 577 characters per sample. For this reason, I introduced an additional data preprocessing step in which the given corpus is first randomly partitioned into N fragments of approximately equal size. The fragment size varied between 30 and 40 characters. The fragments are shuffled and then rejoined to simulate a dataset consisting of concatenated small text samples whose content–especially with respect to neighboring fragments in the shuffled corpus–may be unrelated. This was the only meaningful pre-processing applied to the corpora; several haphazard special characters that were inserted as part of the text file encoding were stripped, but casing and punctuation were preserved.

3.4 Model Details

I use a character-level RNN as generative model with which to expand the desired corpora. Specifically, I use a simple single-layer Long Short Term Memory (LSTM) network [Hochreiter and Schmidhuber, 1997], the standard recurrent architecture for processing sequential data with potentially long dependencies. The models all used a hidden layer with 256 neurons with a character-embedding of length 32 and minibatches of size 128. They were trained to minimize the cross entropy loss using the optimizer RMSProp. Note that the *perplexity* P–a standard information-theoretic measure of how well a sample text is modeled by an underlying distribution is defined as

$$P = 2^{H(p,q)} = 2^{-\sum_x p(x) \log q(x)},$$

where p denotes the 'true' character distribution and q the predicted distribution. $H(p, q)$ is the cross-entropy, thus decreasing the loss function corresponded to an exponential decrease in perplexity. A scheduled learning rate decay rate of 10% every 500 time steps was also applied. This learning rate annealing was the only form of regularization employed by the network. These hyperparameter settings were determined using cross-validation. The networks were trained over 12,000 time steps, except in the case where they were trained only on the small target corpora, which was done for 8,000 steps to avoid overfitting. When pre-training, the networks were first trained for 9,600 time steps on the 'known' corpus, and then for just 2,400 steps on the 'unknown' corpus (an 80%-20% split).

Input Format	HP - Shakespeare	HP - Dutch	HP - Encoded
Trigram Model	34.21	13.15	21.01
Fragmented Small	29.98	7.29	-
Non-Fragmented Small	30.33	7.14	-
Fragmented Full	**25.19**	**2.70**	13.26
Non-Fragmented Full	26.29	2.73	9.85
Fragmented Split	**25.73**	**2.19**	4.86
Non-Fragmented Split	27.18	2.19	7.97

Table 1: Test performance (cross-entropy loss) for HP-Shakespeare, HP-Dutch, and HP-Encoded pairings; note that for the "full" and "small" condition, training is only done on the target coprus; that is Shakespeare, Dutch, and Encoded ($\Lambda = 1000$). **Bold** text indicates results of special note.

Additionally, I train a simple trigram language model on each target corpus as a baseline.

4 Results

4.1 Performance

The LSTM was trained on three pairs of copora (format "Base Text - Target Text"): Harry Potter (HP) - Shakespeare, HP - Dutch, and HP - Encoded, on both fragmented and non-fragmented data. For the encoded English, an inverse ambiguity degree of $\Lambda = 1000$ was used. The trigram model significantly underperforms the neural networks models. The three experimental conditions for the LSTM were "small," indicating that the network was trained only on the reduced target corpus of around 7,800 characters, "full" indicating that the network was trained on a full-sized target corpus of 470,000 characters, and "split," indicating that the network was pre-trained on the 470,000 character base text before being trained on the small target text. Results for test cross-entropy loss are summarized in Table 1. As expected, test performance was greatly improved by moving from the small dataset to the the full dataset, and importantly, pre-training the network in the split condition resulted in performance that, while not better than the full condition, was quite close to it (and a significant improvement over the small condition). More concretely, the percent improvements of the full condition over the split condition for HP-Shakespeare, HP-Dutch, and HP-Encoded were 2%, 18%, respectively. Interestingly, the model pre-trained on English outperformed the model purely trained on the Encoded corpus. While the percent difference grows as the task becomes harder, absolute difference remains relatively constant. Interestingly, training on the fragmented corpora produced consistently better results than training on the non-fragmented datasets. It is also significant to note that these patterns held across corpus-pairs. That is, pre-training on English was nearly as good as training on the full Dutch dataset.

Another experiment was done to investigate the affect of the inverse ambiguity factor on performance on the HP-Encoded. Three values of Λ were tested, 100, 1,000, and 10,000. As one would expect, with a higher Λ, there is less ambiguity

and better performance. The results can be seen in Figure 2.

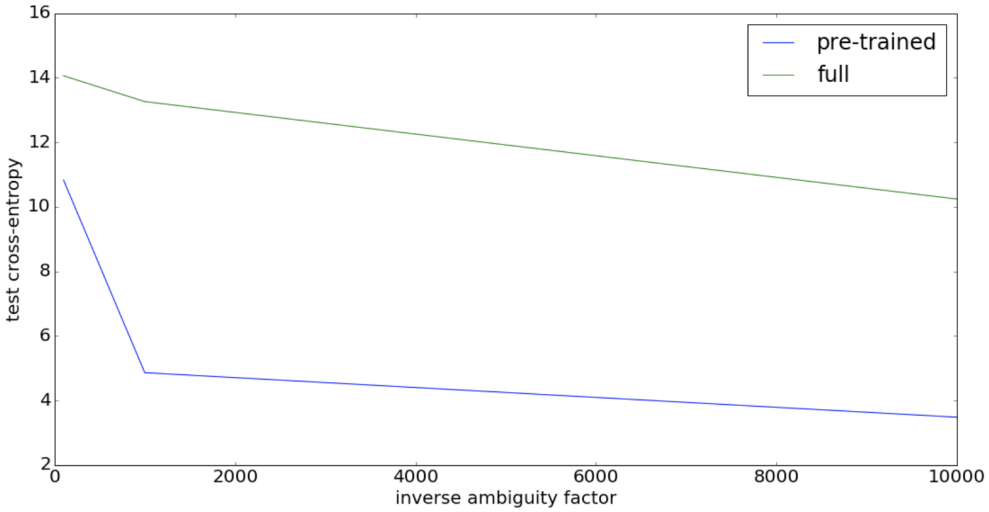

Figure 2: The test performance of the network trained on the HP-Encoded pair for different values of the inverse ambiguity factor Λ

4.2 Novel Text Generation

The next step in the approach was to track the proportion of novel n-grams generated by the networks in the split condition. This proportion was tracked from $n = 2$ to 15 for three 'temperature' setting, a parameter that roughly controls how much variety the network allows when sampling [Stewart, 2016]. A higher temperature results in samples that have much greater linguistic variety but may result in reduced grammaticality and spelling accuracy, while lower temperatures result in text that is much more faithful to the training corpus but lacks diversity. To track the production of novel n-grams, the network generated 5,000 character samples with three temperature settings: 0.75, 1.00, and 1.25. The results are summarized in Figure 3. We can see that, as expected, a higher temperature results in a much greater proportion of novel word sequences, and the proportion of novel sequences increases with n.

4.3 Grammaticality

The final step consisted of tracking, for each temperature setting, the proportion of novel n-grams that could be successfully parsed. As there is no easily available Dutch parser, this step was undertaken only for the HP-Shakespeare corpus pair. For each n-gram length, I tracked the proportion of novel sequences that could be successfully parsed to a sentential constituent. The results are shown in Figure 4.

Interestingly, while as one would expect, the proportion of grammatical sequences decreases with n for a temperature parameter of 0.75, for the other temperature settings it remains relatively constant at a high value.

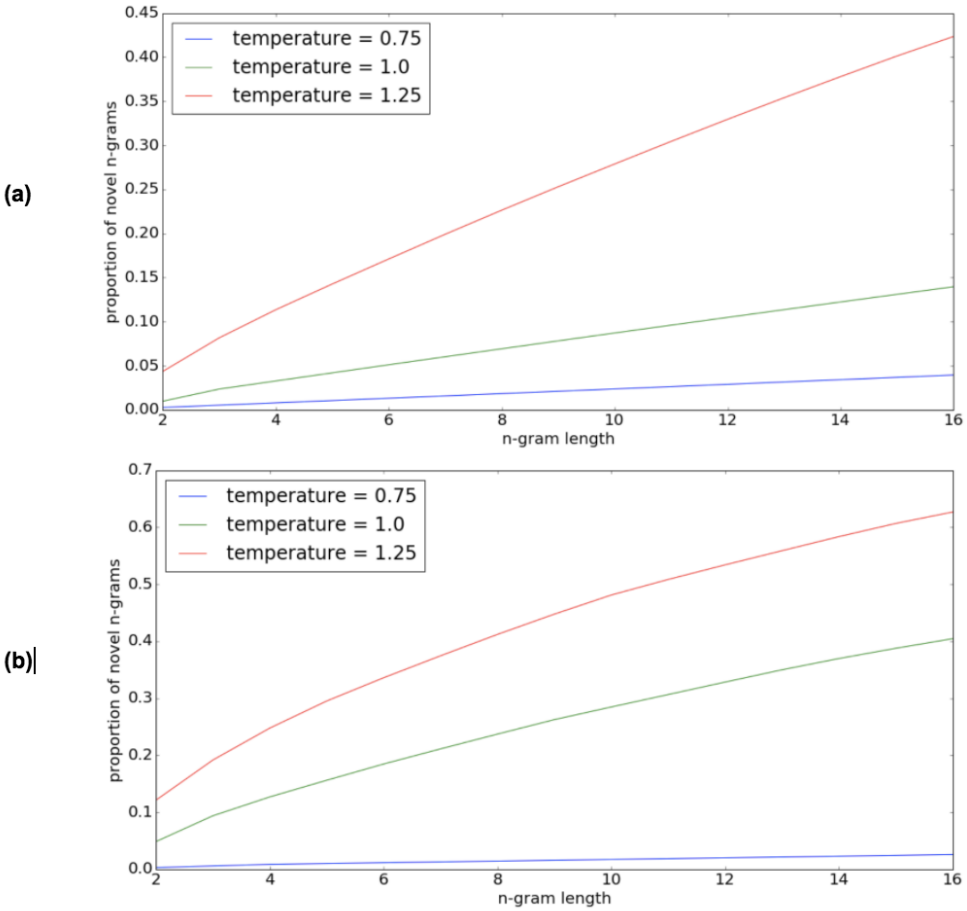

Figure 3: The proportion of novel n-grams generated by the network as a function of n, for n ranging from 2 to 15. (a) is the graph for HP-Shakespeare, and (b) is the graph for HP-Dutch.

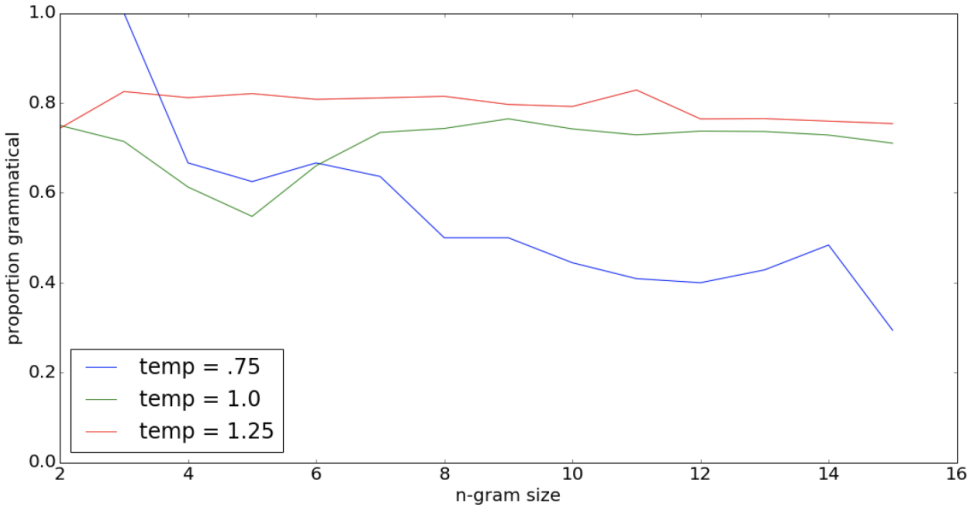

Figure 4: The proportion of grammatical novel n-grams generated by the network as a function of n, for n ranging from 2 to 15.

4.4 Qualitative Evaluation

While quantitative measures of performance are important, it is also necessary in language modeling to observe the actual model outputs. For the HP-Shakespeare pair, a comparison is presented in Figure 5.

(a)

```
ROSS OF YORIA:
Shat cosse, heareth will to thy yes? MENRY Aufigaret up thimereard, hancy spithee, begally, willon if no so thee dam,
his be
Yor iffer 'tized? God she that whice: mir ing fordsto ther no thopeare theave: ar, cou noweards, per goo tave th for
Uposper: hoss: yoursuckint Citiews briburld hat des,
ink
My hent nows rim al IIII:
He him un'd deens rand,
Ares sall'd thy
SICHARD I his tornest alieranownigh lown you anand mas gracque. Ands of heipeadd and hateld, for:
```

(b)

```
First Citizen:
That fered at do't them you mormented to wal 'gry's and Citizen:
We are a maknours incore, I nere cormon, the greivers and do you receik, stells. What staingned is they
and evers.

First Citizen:
Nousl. Aspedese,
Agly cares for dode well, well, you reevem. Iy was; to this guesen pattill food carcever inde

LORD BERKELEY:
My Lord of Derby, many a rebellion, if you I live I will stum.
```

(c)

```
MENENIUS:
Ha! Thou urged myself,
That late but to be brief winged your loyal quest to wail and both
more brothers off that he did thy charge
By using. Come it in.
Give me some were to tell your charge, or no.
```

(d)

```
MARCIUS:
Thanks. What's the matter, you dissentious rogues,
That, rubbing the poor itch of your opinion,
Make yourselves scabs?

First Citizen:
We have ever your good word.

MARCIUS:
He that will give good words to thee will flatter
```

(e)

```
MENENIUS:
There was a time when and clubs:
Rome and her rats are Citizen:
Should by the cormorant belly be why stay we prating here? to revenge this with
our pikes, ere we faults, with surplus, to tire in lowest, basest, poorest,
Of this most wise receive the general food at ne'er came from the lungs, but think we are too dear: your good word.

MARCIUS:
He that will it be done: away, away!
```

Figure 5: Samples of model output: (a) a trigram language model, (b) LSTM trained only on small 7,800 character corpus, (c) LSTM trained on full 470,000 character corpus, (d) LSTM pre-trained on Harry Potter, then on small corpus, and (e) actual Shakespeare from the training set.

Qualitatively, significant jumps in improved sample quality are evident between (a) the trigram model and (b) the LSTM trained on the small corpus, as well as between (b) and (c) the LSTM trained on the full corpus, and (d), the LSTM pre-trained on Harry Potter. It is clear that the LSTM is far better at capturing the general structure of the text than the trigram model, making better use of the 'SPEAKER: dialogue' format present in the training set. Importantly, it is difficult to distinguish the difference between (c) and (d), both of which seem to closely follow (e), the actual Shakespeare.

To visualize the output of the HP-Dutch corpus pairing, I ran a sample output

of the network through a popular multilingual deep-learning based translation tool[2]. The results are shown in Figure 6.

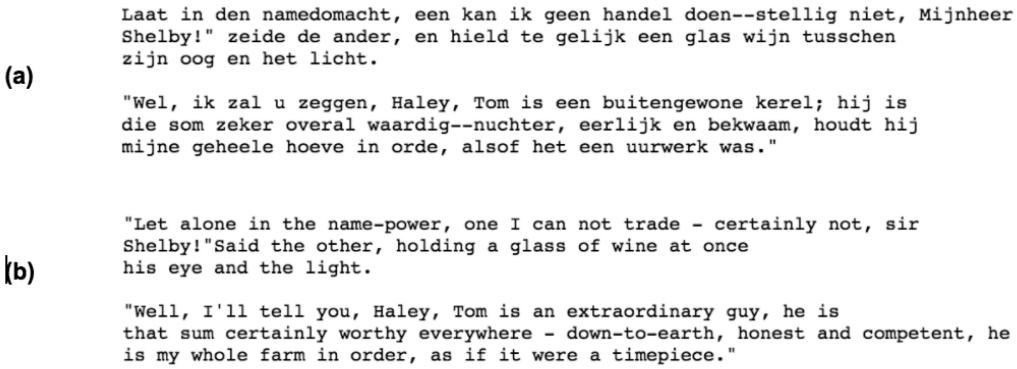

Figure 6: (a) Model output, (b) Dutch-English Translation of (a).

In the author's view, this figure is perhaps the most significant demonstration of the potential of this approach. After training on only English text, merely 2,400 iterations on Dutch writing enables the network to produce exceptionally high-fidelity output.

5 Conclusion

This approach demonstrates promise for continued work and application to real unknown writing systems. Pre-training the model on a related language/writing system produces a clear benefit over training only on a limited corpus, and moreover manages to closely track the performance of networks trained on a full-sized corpus of the target writing system. However, there are several areas for improvement and further investigation.

First, a more discriminative parser should be applied to the analysis of the novel n-gram sequences. The reported portion of 'grammatical' sequences above is certainly too high, as inspection of several sequences shows. (For example, "arms too MENENIUS : Why , masters , mine honest neighbours ," was parsed as a complete sentence.) While the Stanford parser is highly accurate with grammatical sentences, it was designed to find parsers on input for which it was expected there would be a correct parse–it is not a grammar checker, and therefore tries very hard to find a parse, only reluctantly labeling an input as a fragment. It is difficult to gauge the success of step 3 in my approach without a more stringent parser. This is also important to determine whether higher sampling temperatures really do result in more grammatical output, which is generally not the case. However, it is possible that fragmenting the corpus has some effect on in the inherent diversity of the training text.

Second, the model used here was a very basic single-layer LSTM. While it may be the case that the smaller architecture and relatively short training time allowed the model to learn the structure of the main text while not overfitting and being unable to transfer to the target text, more complicated, deeper architectures are

[2]Google Translate

worth exploring. Other generative deep learning methods, such as variational autoencoders (VAEs) [Kingma and Welling, 2013] or generative adversarial networks (GANs) [Goodfellow et al., 2014] might be applied to similar effect.

Third, it would be a better approximation of the fragmented nature of many extant corpora if the fragmentation process were done across texts, not just within one text. That is, if fragments were drawn from a variety of sources, and thus truly unrelated. However, this is likely not a major factor, as many of the extant text samples of ancient languages, though distinct, deal with similar subject matter (farming, trading inventories, etc.).

Fourth, an important next step for the method would be to test it on logographic writing systems, i.e. Chinese. Many ancient writing systems have a pictographic component (thought not to the same degree as is commonly thought; for example Ancient Egyptian and Mayan hieroglyphs are both primarily syllabic). In keeping with this theme, I am currently exploring the option of training a convolutional network to visually classify the underlying symbols in logograms, i.e. to identify whether an abstract character is based on the sun or a person, for example. If sufficient performance is achieved, this network could be applied to characters in unknown writing systems as a means of determining meaning.

Fifth, it is worth investigating the positive effect that fragmenting the corpus appears to have on performance. For both the HP-Shakespeare and HP-Dutch pairs, the network trained on the fragmented data outperforms that trained on the unaltered texts. This could be because fragmentation allows the network to view more diverse content from the input together, enabling it to develop a more balanced distributional representation of the character dependencies, and because the data is shuffled within-text (and not between different texts), the fragments are not so different that it impedes learning. However, more work on this is needed, as it could prove to be a more broadly applicable method of pre-processing for the application of deep nets to natural language.

Finally, continuing to refine the application of the technique to numerically-encoded English would likely yield important insights into the impact of character-level ambiguity on decipherment.

From a purely machine learning perspective, the main contribution of this work is the demonstration that pre-training a character-level RNN on any text with a similar structure can result in generated text that is nearly as good as if it the model were trained entirely on the target text. The reasons behind this adaptability are interesting topics for future exploration. For example, is this ability more closely linked to the flexibility of these network architectures and their power as learning machines? Or is it based on underlying similarities in language structure, relating to the idea of a Universal Grammar [Chomsky and George, 1990]? This flexibility could also be applied to other areas of natural language processing. For example, in machine translation, it is often difficult to develop models to translate to low-resource languages, but if an RNN could be pre-trained on a language related to a low-resource language, and then on whatever translation data is available, it could offer a marked improvement. Thus, this method offers not only new possibilities for decipherment, but potentially new understanding and applications for deep learning more broadly.

6 Bibliography

References

Linear a, 2015. URL http://minoan.deaditerranean.com/linear-b-transliterations/knossos/kn-d-2/kn-dl/.

Project gutenberg, 2017. URL https://www.gutenberg.org/.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 153–160. MIT Press, 2007. URL http://papers.nips.cc/paper/3048-greedy-layer-wise-training-of-deep-networks.pdf.

John Chadwick. *The decipherment of linear B*. Cambridge Univ. Press, 2014.

Noam Chomsky and Alexander George. *Reflections on Chomsky*. B. Blackwell, 1990.

Michael D. Coe. *Breaking the Maya code*. Thames & Hudson, 2012.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann LeCun. Very deep convolutional networks for natural language processing. *CoRR*, abs/1606.01781, 2016. URL http://arxiv.org/abs/1606.01781.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, page 625–660, 2010.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

Alex Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013. URL http://arxiv.org/abs/1308.0850.

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006. doi: 10.1162/neco.2006.18.7.1527.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv*, 2013.

Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - ACL 03*, 2003. doi: 10.3115/1075096.1075150.

Kevin Knight, Anish Nair, Nishit Rathod, and Kenji Yamada. Unsupervised analysis for decipherment problems. *Proceedings of the COLING/ACL on Main conference poster sessions* -, 2006. doi: 10.3115/1273073.1273138.

Tomas Mikolov and Geoffrey Zweig. Context dependent recurrent neural network language model. *2012 IEEE Spoken Language Technology Workshop (SLT)*, 2012. doi: 10.1109/slt.2012.6424228.

Maurice Pope. *The story of archaeological decipherment: from Egyptian hieroglyphs to Linear B*. Scribner, 1975.

Igor Pozdniakov and Konstantin Pozdniakov. Rapanui writing and the rapanui language: Preliminary results of a statistical analysis. *Forum for Anthropology and Culture*, 2007.

Marc Aurelio Ranzato, Ylan Boureau, and Yann L. Cun. Sparse feature learning for deep belief networks. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1185–1192. Curran Associates, Inc., 2008. URL http://papers.nips.cc/paper/3363-sparse-feature-learning-for-deep-belief-networks.pdf.

Andrew Robinson. *Lost languages: the enigma of the world s undeciphered scripts*. McGraw-Hill, 2002.

J K Rowling. *Harry Potter and the Sorcerer's Stone*. Scholastic Corp., 1998.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy P. Lillicrap. One-shot learning with memory-augmented neural networks. *CoRR*, abs/1605.06065, 2016. URL http://arxiv.org/abs/1605.06065.

Benjamin Snyder, Regina Barzilay, and Kevin Knight. A statistical model for lost language decipherment. *ACL '10 Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, page 1048–1057, 2010.

Russell Stewart. Maximum likelihood decoding with rnns - the good, the bad, and the ugly, 2016. URL https://nlp.stanford.edu/blog/maximum-likelihood-decoding-with-rnns-the-good-the-bad-and-the-ugly/.

Harriet Beecher Stowe. *De Negerhut*. 1852.

Ilya Sutskever, James Martens, and Geoffrey Hinton. Generating text with recurrent neural networks. *Proceedings of the 28th International Conference on Machine Learning*, 2011.

Oriol Vinyals, Charles Blundell, Timothy P. Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. *CoRR*, abs/1606.04080, 2016. URL http://arxiv.org/abs/1606.04080.

Nisha Yadav, Hrishikesh Joglekar, Rajesh P. N. Rao, Mayank N. Vahia, Ronojoy Adhikari, and Iravatham Mahadevan. Statistical analysis of the indus script using n-grams. *PLoS ONE*, 5(3), 2010. doi: 10.1371/journal.pone.0009506.

Kenji Yamada and Kevin Knight. A computational approach to deciphering unknown scripts. *Proceedings of the ACL Workshop on Unsupervised Learning in Natural Language Processing*, 1999. doi: 10.3115/1073012.1073079.

John Younger. Linear a texts and inscriptions in phonetic transcription, 2000. URL http://www.people.ku.edu/~jyounger/LinearA/#5.