Computational Pattern Recognition in Linear A

Jajwalya R Karajgikar, Amira Al-Khulaidy, Anamaria Berea

Computational Pattern Recognition in Linear A

Jajwalya R. Karajgikar[1*], Amira Al-Khulaidy[2], Anamaria Berea[3]

1 Computational Sciences, MS, George Mason University

2 Computational Social Sciences, PhD student, George Mason University

3 Computational and Data Sciences, Associate Professor, George Mason University

*Corresponding author: Jajwalya R. Karajgikar[1]

Abstract

Linear A is an ancient Mycenaean Greek language that is as of yet undeciphered. Computational analysis of the symbols using natural language processing and data mining techniques can aid to better understand this logogram language, and to help us uncover interesting statistical and information-theoretic patterns even without deciphering the language. Using information theory and data science techniques, we show an exploratory analysis of these symbols. We use n-gram analysis to summarize and predict the possible symbols on lost stone tablets, while symbols are clustered by topic modeling as well as k-means, for comparison. The results show that there are universal features in this language that we can explore further with computational methods.

keywords

Linear A, Information Theory, NLP, Topic modeling, Clustering, Knowledge Mining, Humanities and Social Sciences/Literature, Humanities and Social Sciences/History

INTRODUCTION

The historical context of the ancient languages may be lost to time but some of the archaeological evidence found in the form of stone tablets (as in our case), contain information that can be considered for the possibility of computational study. In the case of undeciphered languages, such as Linear A, many researchers turned to new computational techniques such as machine learning and natural language processing with the aim to decode them. But fewer authors have approached these languages from the data science and information theory perspective, with the aim to discover statistical and informational patterns which, even in the absence of a decipherment, would help us understand better the universality of human language, both ancient and modern.

From the data science perspective, languages are repositories of information that hold quantitative value in addition to the meaning that we are assigning them from the human cultural perspective. Undeciphered languages represent a type of agnostic data that can help us design better, unbiased algorithms, particularly for natural language processing methods. Therefore, in this paper, our main goal is to extract meaningful emergent patterns of vocabulary from an independent primary source, Linear A.

British Archeologist Arthur Evans discovered the hieroglyphic seal-stones with successive types of script around 1893 along the eastern Mediterrennean basin, which culminated in the dramatic excavation of the palace hill of Mycenaean Knossos on the Greek island of Crete (Evans, 1909). Dated from around 1400 BCE, the ancient Minoan civilization inscriptions are one of the earliest

[1] jkarajgi@gmu.edu / jajwalyak@gmail.com

forms of writing discovered. Evans was the first to define the pictographic Cretan hieroglyphics, and the subsequent linear of classes A and B. The unique structure of the ancient palace site was such that Evans believed the possibility of tracing the whole evolution of the art of writing all the way up to the Neolithic times, thereby establishing its importance in historical linguistics and art history. The Linear A writing system was prevalent in the Bronze Age Minoan civilization from 1800 to 1450 BC. It was the primary script used in palace and religious writings of the Minoan civilization. The etchings on the clay tablets found in the excavation are called logogram symbols[2] and form the basis of our dataset that will be run through several analysis techniques before they are available for subject matter expertise' discernment and approval. This research work is an attempt at taking a step on the identification, simulation, and study of language families origins and evolution with constellations of computational methodology.

I RELATED WORKS

1.1 Literature Survey

While Linear A is as yet undeciphered, the Linear B syllabary has been computationally considered (Papavassiliou et al., 2020) via smoothed n-gram analysis, Hidden Markov Models, Bayesian classifiers, and Conditional Random Fields (CRF). This paper proposes to conduct an exploratory approach to conducting data mining on an undeciphered language. Linear B has been challenging to decipher due to reasons similar to Linear A's. There is a limited amount of data that can be used for processing since there are only so many archaeological remains from the historical evidence of the Linear A language, which makes it a greater challenge to decipher than Linear B. Furthermore, there is no real way of knowing if the output will be accurate without validation data or material.

A machine-learning system capable of deciphering lost languages (Luo et al., 2020) was demonstrated by having it decipher Linear B—the first time this has been done automatically. The approach used was very different from the standard machine translation techniques by using a novel neural decipherment approach for automatic decipherment. The outcome accuracy can be verified and validated against existing Linear B literary decipherment.

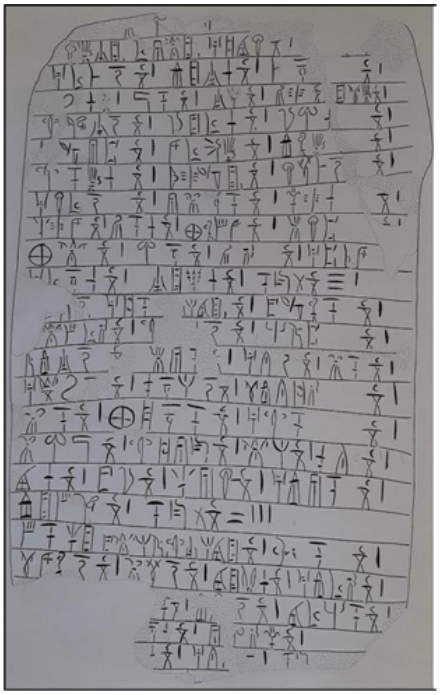
In the below Linear B tablet facsimile[3] (see figure 1, right), the following transliterations[4] are known: men (ideogram[5] VIR), women (ideogram MUL), girls (ko-wa) and boys (ko-wo). This is available because the women and children of Knossos were identified by the occupational, geographical, or ethnic names. The DĀMOS or Database of Mycenaean at Oslo (Aurora, et al., 2013) is a corpus of complex architecture just for Linear B syllabary that is standardized by the TEI epigraphic standard and Leiden convention that contains 5900 documents of metadata, textual markup, annotation, and modularity in a relational database MySQL workbench. The purpose is to translate each lemma, disseminate knowledge of the language, and its irregularities.

[2]In a written language, a logogram or logograph is a written character that represents a word or morpheme.

[3]A facsimile is a copy or reproduction of an old book, manuscript, map, art print, or other item of historical value that is as true to the original source as possible.

[4]A transliteration is a transcription from one alphabet to another transcription, written text - something written, especially copied from one medium to another, as a typewritten version of dictation.

[5]An ideogram is a character or symbol representing an idea or a thing without expressing the pronunciation of a particular word or words for it, as in the traffic sign



]du-ru 1 ko[
ko-no-si-ja , ra-wa-ke-ja , a-nu-wi-ko VIR 1[ ]*vestigia*[
a-ra-da-jo VIR 1 pi-ja-si-ro VIR 1 da-na[ ] VIR 1
]we-ro VIR 1 po-to VIR 1 si-pu2 VIR 1 pu-te VIR [1] ja-sa-no VIR 1
qa-me-si-jo VIR 1 mi-ja-ra-ro VIR 1 mi-ru-ro VIR 1
[.]-ki-wa-ta VIR 1 u-ra-mo-no VIR 1 pi-ri-no VIR 1
qa-to-no-ro VIR 1 pe-te-ki-ja , VIR 1 ko-ni-da-jo VIR 1
a-ko-ra-jo VIR 1 wa-du-[.]-to VIR 1 qo-te-ro VIR 1
i-te-u VIR 1 pu-to-ro VIR 1 ka-ri-se-u VIR 1 a3-ko-ta VIR 1
ka-ke VIR 1 ru-na VIR 1 pu-wo VIR 1 a-ta-ze-u [VIR 1
a-ra-na-ro VIR 1 si-ja-pu2-ro VIR 1 to-so x VIR 31
]-ti-jo , a-nu-to [qa-]si-re-wi-ja , VIR 1 su-ki-ri-to VIR 1
[.]ke-se-ra-wo VIR 1 qa[. .]-jo VIR 1 ne-o-ta VIR 1
o-pi-si-jo VIR [1] a3-wa-to VIR 1 a-ti-jo VIR 1 du-to VIR 1
a-de-we-[.] VIR 1 pa-na-re-jo VIR 1 sa-pi-ti-nu-wo VIR 1
du-to VIR 1 ka-*56-na-to VIR 1 a-qa-to VIR 1
du-ru-po VIR 1 qa-nu-wa-so VIR 1 ke-re VIR 1 pa-ti VIR 1
wi-ro VIR 1 su-mi VIR 1 i-wa-ko VIR 1 a-pi-wa-to VIR 1
pi-ja-se-me VIR 1 to-so x VIR 23
se-to-i-ja su-ke-re-o , qa-si-re-wi-ja VIR 1 ku-to VIR 1
sa-u-ri-jo VIR 1 du-ni-jo VIR 1 wi-ja-ma-ro VIR 1 a-pi-ra-wo VIR 1
]to VIR 1 pi-[.]-jo VIR 1 wi-ra-ne-to VIR 1
]na-ro VIR 1 wa-ru-wo-qo VIR 1
] VIR 1 a-ti-[ ] VIR 1 to-so[
]vest.?[

Figure 1: Left, the Mycenaean Tablet KN As 1516 (Chadwick, et al., 1987). Right, the transliterated text of the KN As 1516 (Chadwick, et al., 1987)

There are rich interdependencies between various unknown languages and their applicability in unraveling mysteries of modern evolutionary biology studies (Daggumati et al., 2019). Genomic data availability lies in the gray area of abundance and sparseness depending on the kind of algorithm at work. The construction of phylogenetic trees uses algorithms like Neighbor joining method, Maximum Likelihood, Common Mutations Similarity Matrix, which are comparatively objectively free of human bias (Daggumati et al., 2019). The ClustalW2 cladogram with 2 phylogenetic tree generation algorithms using both Neighbour Joining method and UPGMA method (unweighted pair group method with arithmetic mean) give the similarity index between various comparable language models.

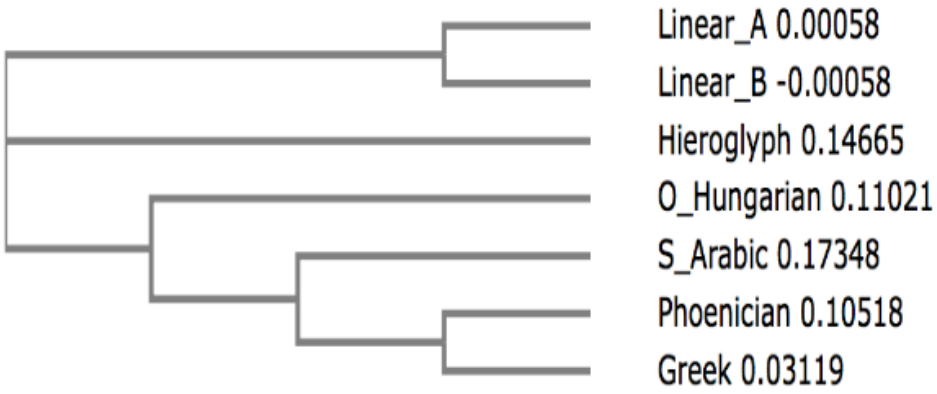


*Figure 2: Evolutionary Tree of ancient scripts reconstructed using Neighbor Joining algorithm in ClustalW2 (*Daggumati et al., 2019*)*

As such, languages with the same proto-language structures having similar distribution profiles is not unknown to linguists. The sequence-to-sequence model capturing character-level correspondence between cognates using minimum cost flow problem, that is Long Short Term Memory algorithm. There are many more research papers to consider going forward but these provide a guideline to previous methodology and are indicative of the expected results.

1.2 Research Question(s)

Can we use natural language processing and data mining methods to create a meaningful taxonomy of vocabulary from a sparse dataset showing patterns of a language that is yet to be deciphered? To specify, the question is: What are the frequently occurring patterns in the Linear A corpus that are also similar in form?

1.2.1 Objectives

- Computational linguistic preprocessing and analysis via vector embeddings and similarity analysis via K-Means Clustering.
- Methods for the detection of intertexts and text patterns using Zipf Law and Entropy possibilities.

1.2.2 Motivation

We are interested in this particular problem for its applications in historical Computational Linguistics research. The multidisciplinary nature of the project is exciting for gaining knowledge in several domains in Digital Humanities. Furthermore, algorithms used with Natural Language Processing are applicable in a variety of fields like Evolutionary Biology Genomics. This is part of the wider project on studying the origins of socio-linguistic complexity combining natural language processing, information theory, data science techniques, and eventually even agent-based modeling.

II Model

2.1 Dataset

2.1.1 Preparing the Data

The Linear A data was available in JSON format in the Github repository (mwenge / lineara.xyz, 2019). The original .ods file was converted into a series of .csv extension files for easy machine readability, based on each attribute in the ODS extension file[6]. To read the inscriptions in Linear A and Linear B, special unicode font character installations are required: NotoEmojiRegular.ttf[7], NotoMono-Regular.ttf[8], NotoSansLinearA-LinearB.ttf[9]. When text is

[6] A file with the .ODS file extension is most likely an OpenDocument Spreadsheet file that contains spreadsheet information like text, charts, pictures, formulas and numbers, all placed within the confines of a sheet full of cells.
[7] https://github.com/googlefonts/noto-emoji
[8] https://github.com/googlefonts/japanese/tree/main/build/fonts/noto
[9] https://github.com/googlefonts/noto-fonts

rendered by a computer, the characters in the text can not all be displayed, because no font that supports them is available to the computer. When this occurs, small boxes called “tofu” are shown to represent the characters. The Python package Matplotlib also does not render the font as of the current version 3.3.4.

2.1.2 Dataset Schema

Now the files can be opened to show the attribute features:
Name, Parsed Inscription, Transcription, Translated Words

	#	source	name	normal copy	line	sign groups	ideograms	integers	fractions	probable “words”
0	1.0	8.052	AN Zb 1	𐘀	None	𐘀	None	None	None	𐘀
1	2.0	4.006	AP Za <3>	𐘁𐘂𐘁	None	𐘁𐘂𐘁	None	None	None	𐘁𐘂𐘁
2	3.0	4.002	AP Za 1	𐘃𐘄𐘅𐘆𐘇𐘂𐘁	None	𐘃𐘄𐘅𐘆𐘇𐘂𐘁	None	None	None	𐘃𐘄𐘅𐘆𐘇
3	4.0	4.004	AP Za 2	𐘈𐘉𐘊𐘋𐘌𐘍𐘂𐘁𐘎𐘏𐘐𐘑𐘒𐘓𐘔𐘂𐘁	1	𐘈𐘉𐘊𐘋𐘌𐘍𐘂𐘁𐘎𐘏𐘐𐘑𐘒𐘓𐘔	None	None	None	𐘈𐘉𐘊 𐘋𐘌𐘍𐘂𐘁 𐘎𐘏𐘐𐘑𐘒𐘓𐘔
4	5.0	4.004	AP Za 2	𐘕𐘖𐘗𐘘𐘙𐘚𐘛𐘂𐘁𐘜	2	𐘕𐘖𐘗𐘘𐘙𐘚𐘛𐘂𐘁𐘜	None	None	None	𐘕𐘖𐘗 𐘘𐘙𐘚𐘛𐘂𐘁 𐘜
5	6.0	4.142	AR Zf 1	𐘞𐘟𐘠	None	𐘞𐘟𐘠	None	None	None	𐘞𐘟𐘠
6	7.0	4.143	AR Zf 2	𐘞𐘟𐘠𐘁𐘂	None	𐘞𐘟𐘠𐘁𐘂	None	None	None	𐘞𐘟𐘠𐘁𐘂
7	8.0	8.048	ARG Zg 1	𐘡 ≈ 𐘢𐘣	None	𐘡 ≈ 𐘢𐘣	None	None	None	𐘡 ≈ 𐘢𐘣

Figure 3: Linear A Dataset from Github repository [mwenge](#) / [lineara.xyz](#)

2.1.3 Preprocessing the Data

The first step in the preprocessing pipeline is the fragmentation of the text blocks into sentences. However, in the case of our dataset, the values are in dataframe format and not in larger blocks of text. It is possible to consider the possible sentence boundaries depending on the transliterated pieces.
During the initial steps of the pre-processing pipeline, the content is fragmented into sentences. In numerous dialects and languages -- for example, English -- accentuation, particularly the full stop or period character, outcry ,and question marks can be utilized to distinguish the finish of the sentence. However in the case of our dataset, we are unsure of the actual sentence boundaries since tablets are not indicative of word tokenization although it has been considered within the *probable “words”* column of the dataset. This is a major challenge in both processing the data and deciphering the language in general.

2.2 System Framework

2.2.1 Exploratory N-Gram Analysis

The purpose of this module is to explore the Linear A dataset for better understanding of the distribution of symbols in the corpus. We implemented different NLP techniques such as generating bigrams, trigrams, predicting symbols based on weighted frequency and generating a summary of the dataset. The diagram (see Figure 4) shows the process flow of this module. The process starts with importing required Python libraries nltk. The bigrams and trigrams collections

are collocations that help us learn of the summary and predicted symbols in the missing or lost stone tablets. Calculating the word weighted frequency generates a summary of the dataset and predicting symbols.

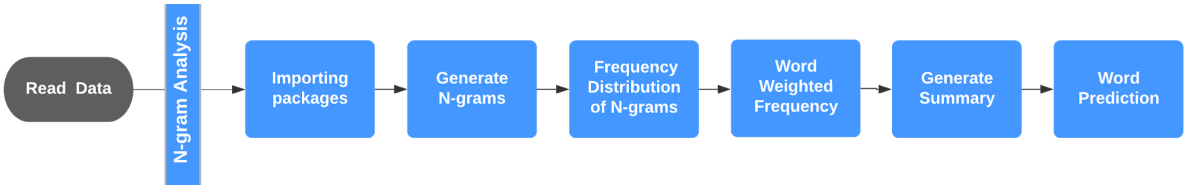


Figure 4: Natural Language Processing N-gram analysis process flow

The frequency distribution of the symbols gave us an interesting Lexical Dispersion plot[10] (see Figure 5). The frequency of a word across the corresponding parts of the corpus highlights the word offsets.

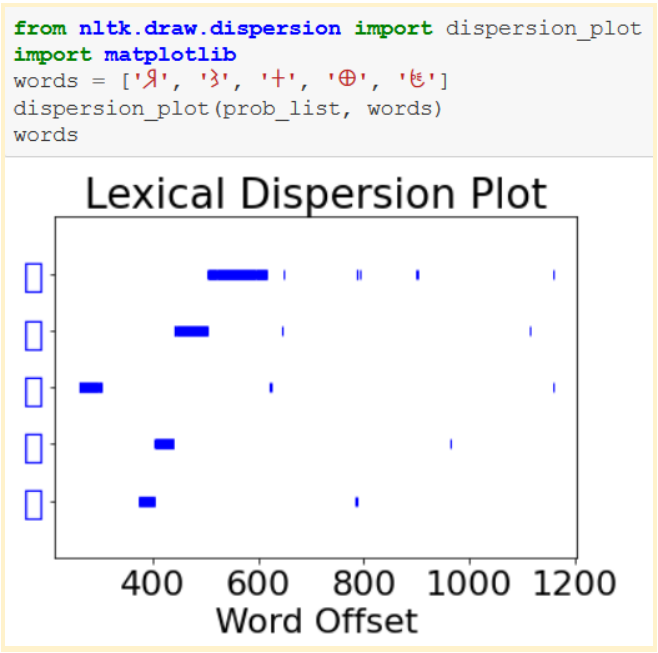


Figure 5: Lexical Dispersion plot for the most frequently occurring symbols

The plot of frequency of the terms is a short tail distribution, that is universal with Zipf's Law[11] parameters (see Figure 6). The terms considered for summary using n-gram analysis where (n-1) terms give us the nth terms, are thus the more highly frequent symbols through the relative frequency table. We find that, interestingly, the predicted symbols are also the ones found at the beginning of other tablets.

[10]Lexical dispersion is a measure of how frequently a word appears across the parts of a corpus. This plot notes the occurrences of a word and how many words from the beginning of the corpus it appears (word offsets).
[11]Zipf's law is an empirical law formulated using mathematical statistics with related discrete power law probability distributions. The underlying principle in Zipf's law accepting models is that short sequences have a high probability of occurring while long sequences have a low probability and hence a high energy (Zipf, 1936).

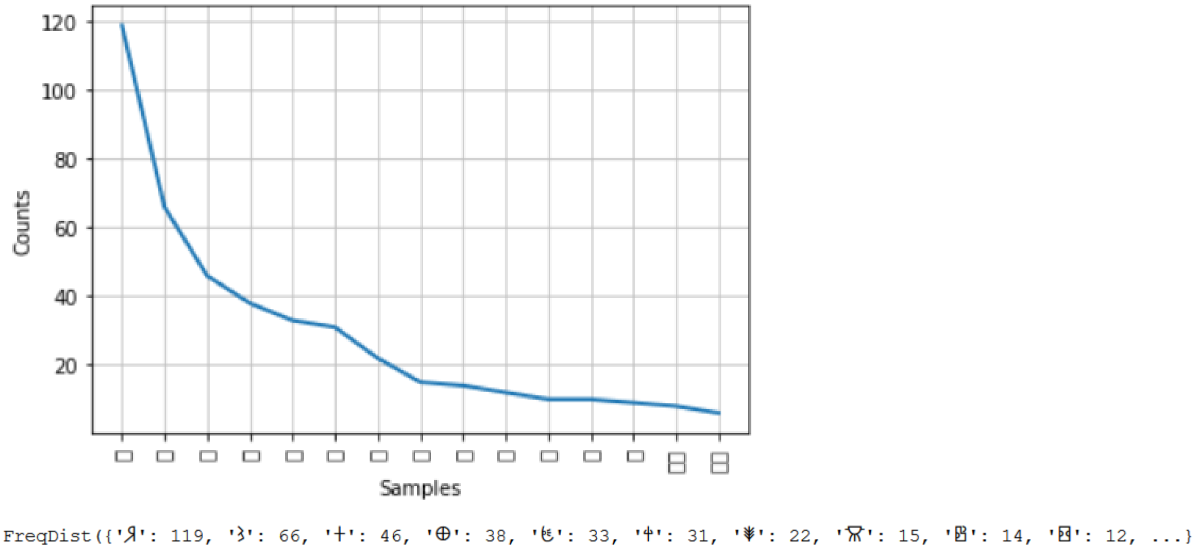


Figure 6: Short-tailed distribution plot of the most frequently occurring symbols

Then we test the Zipf law by plotting first the frequency of the symbol ranked in descending order, then taking the logarithm of these values. Given that this dataset is allegedly 90% administrative records with billing records, the summary of the “goat” symbol along with fractions is an apt summary, we believe, for the dataset in question.

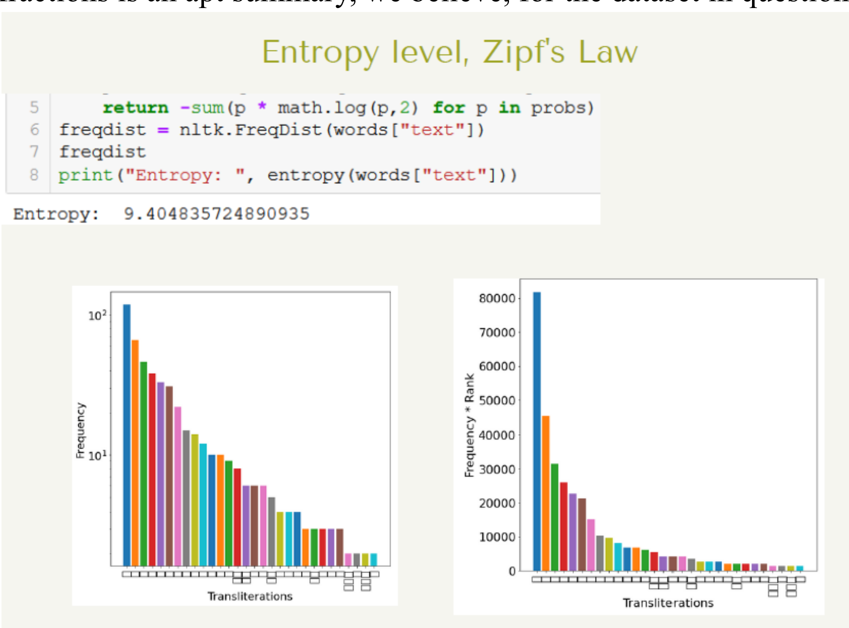


Figure 7: Plotting Zipf Law short-tailed distribution of probable words in the data set

For Entropy analysis, we expected the output value to be comparable to other known languages. The stone tablets containing only 96 unique symbols or transliterations transfers to the entropy[12] analysis where we gain the high entropy level of slightly above 9.4. The short tail distribution of the log of frequency and rank versus the transliterations is also very evident at each iteration.

[12]Entropy is the measure of uncertainty, randomness, or a disorder of a system (Clausius, 1865)

In this case, our analysis of entropy reveals a larger level of disorder in the Linear A language. If the disorder of the symbols is high, that could mean that it is not a language as we understand human languages to be, since most human languages have a similar entropy value (Shannon, 1949). The higher the Shannon entropy[13] value, the greater the disorder and, subsequently, the less likely we are to think of that dataset as a language.

In this case, we obtained two results. The first result was obtained using only the "probable words" from the corpus, instead of the entire corpus. The output revealed a greater entropy of 9.404, than that of languages we know, which was inconclusive in terms of defining whether what we were looking at was a human language. However, when including more of the symbols and ideograms, the entropy decreased to 3, showing that the disorder had been greatly reduced (Matricciani, 1994).
One possible conclusion is that the introduction of more symbols in the dataset allowed the algorithm to recognize more patterns, which reinforces that we are looking at a human language from the information theory perspective (see Figure 8).

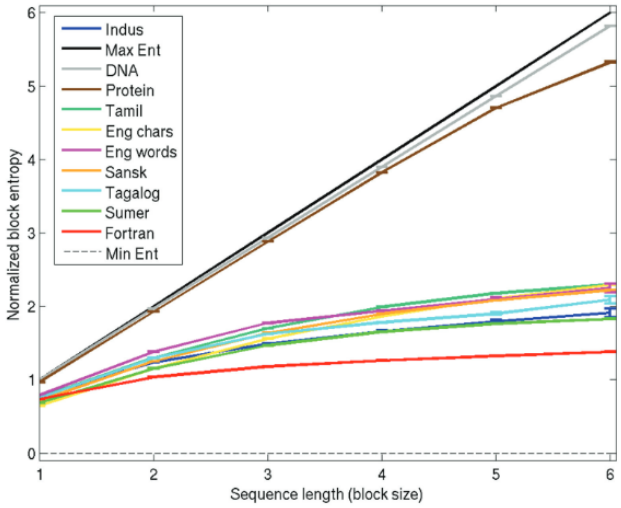


Figure 8: Entropy of natural languages and other sequences. (Rao, 2018)

We trained the Word2Vec model for one epoch and we fit the same model to look at the distribution of the symbols in the corpus, visualized with a scatter plot.

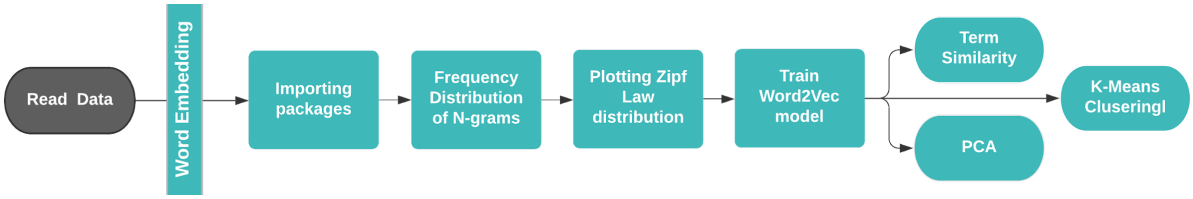


Figure 9: Word Embedding Process Flow

[13]Shannon Entropy formula where it is defined as "In information theory, the entropy of a random variable is the average level of "information", "surprise", or "uncertainty" inherent in the variable's possible outcomes." (Shannon, 1949)

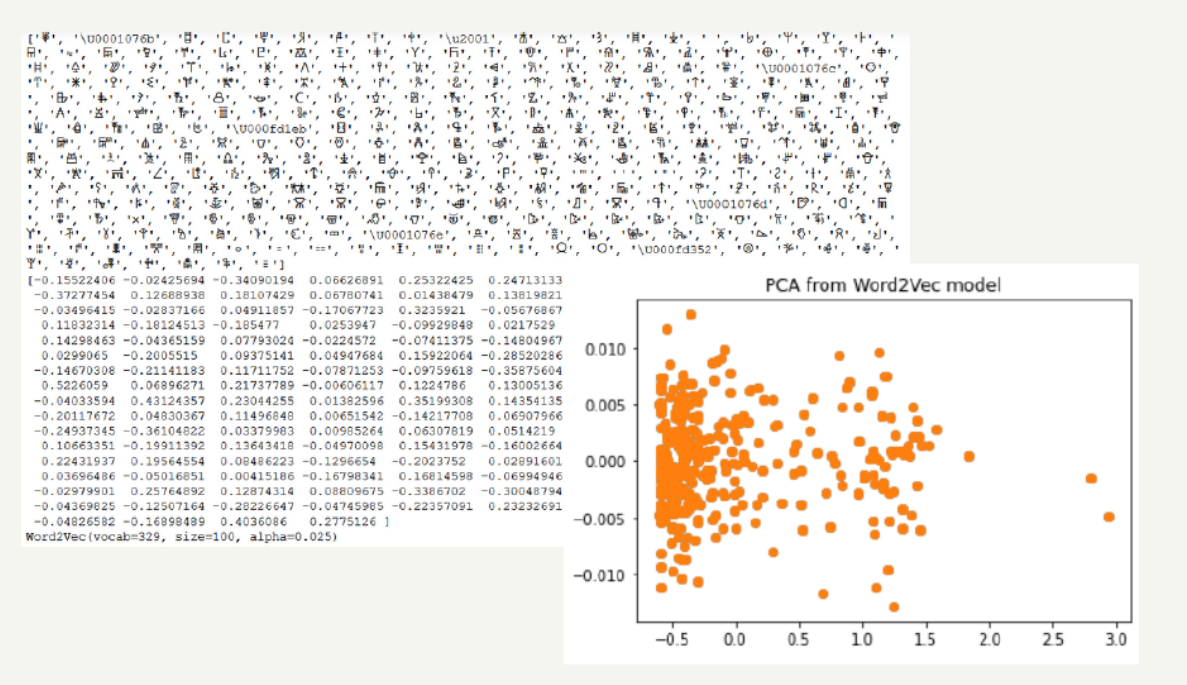


Figure 10: Word Embedding and Principal Component Analysis of Linear A data set

Word embedding of the transliterations show us the entropy value to be of less structured format, as expected with such a sparse dataset. The pictographic representation also leaves much to the randomness. The stochastic gradient descent representation of the covariance and correlation matrix for the PCA shows the dimensionality for a critical value of the alpha at a default number of 0.025. There are certain outliers which are further explored with Topic Modeling and k-Means clustering.

```
print (model.similarity('𐘀', '𐘀'))
print (model.similarity('𐘊', '𐘀'))
print (model.most_similar(positive=['𐘊'], negative=[], topn=4))
print (model.most_similar(positive=[], negative=['𐘀'], topn=4))
```

```
1.0
0.9994925
[(' ', 0.999587893486023), ('\u2001', 0.9995784759521484), ('𐘎', 0.9995618462562561), ('𐘏', 0.9995594620704651)]
[('𐘁', -0.9992014169692993), ('𐘇', -0.9994107484817505), ('𐘊', -0.9994924664497375), ('𐘅', -0.9995063543319702)]
```

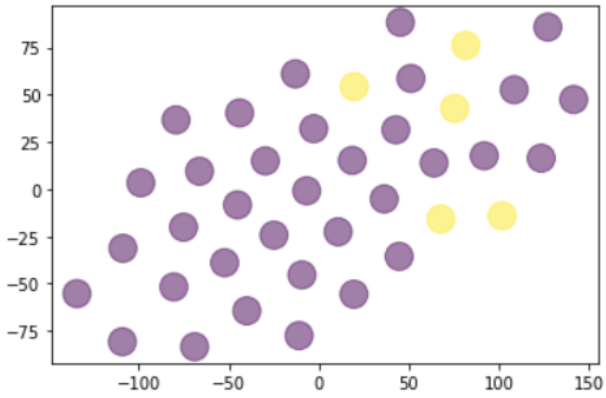
Figure 11: Cosine Similarity analysis of the symbols

Word2Vec embedding also gives us the similarity analysis of the symbols by their vector values. As seen in the output, the first set of symbols are the same, so their similarity comes to 1.0, while the similar but not same symbols values are 0.99. The list of symbols with similar positive values as the first set of symbols and then the negative associations of the second symbol in the second set is also apparent.

```
Score (Opposite of the value of X on the K-means objective which
is Sum of distances of samples to their closest cluster center):
-0.22897068
Silhouette_score:
0.48115668
```

Figure 12: K-means clustering scores

K-Means Clustering gives us the groups of the transliterations based on their cosine similarity distance. The silhouette score is then grouped by the simpler Euclidean distance metric.
When the number of clusters =2,



When the number of clusters =5,

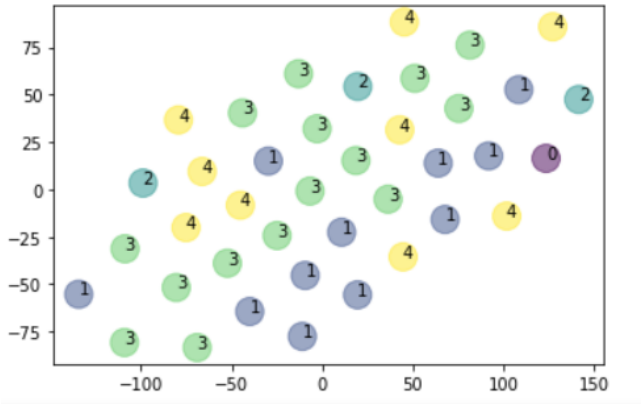


Figure 13: K-means clustering by groups

The within-cluster sum of squares is a measure of the different types of transliterations within each cluster. The average distance from observations in this graph is seemingly random which can be attributed to the high entropy given that it is a measure for the variability of the transliterations within each cluster. Centroid helps interpret each cluster as a focus for the cluster.

As a generative statistical model, Latent Dirichlet Allocation (LDA)[14] uses known observations to group similar unobserved groups of terms, which is extremely useful for undeciphered languages.

[14] The Latent Dirichlet Allocation (LDA) is used to generate a probabilistic model of a corpus with a variety of topics using Jensen–Shannon divergence between topics computation and term frequency relevance. LDA is a useful technique within the machine learning toolbox and thus, the artificial intelligence toolkit.

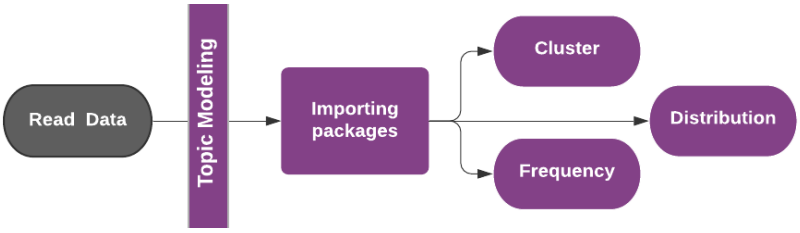


Figure 14: Topic Modeling process flow

LDA was originally developed for Evolutionary biology, bio-medicine, and engineering problems, and in computational linguistics as well.

A word cloud gives the visual representation of most common stone tablets in the text corpus (see Figure 15)

Figure 15: Word cloud of locations the Linear A tablets were found by the number of symbols gathered from each

The stone tablet “HT” seems to be the most popular in the text corpus which corresponds to the known archaeological study of stone tablets being found mainly in Hagia Triada (HT).

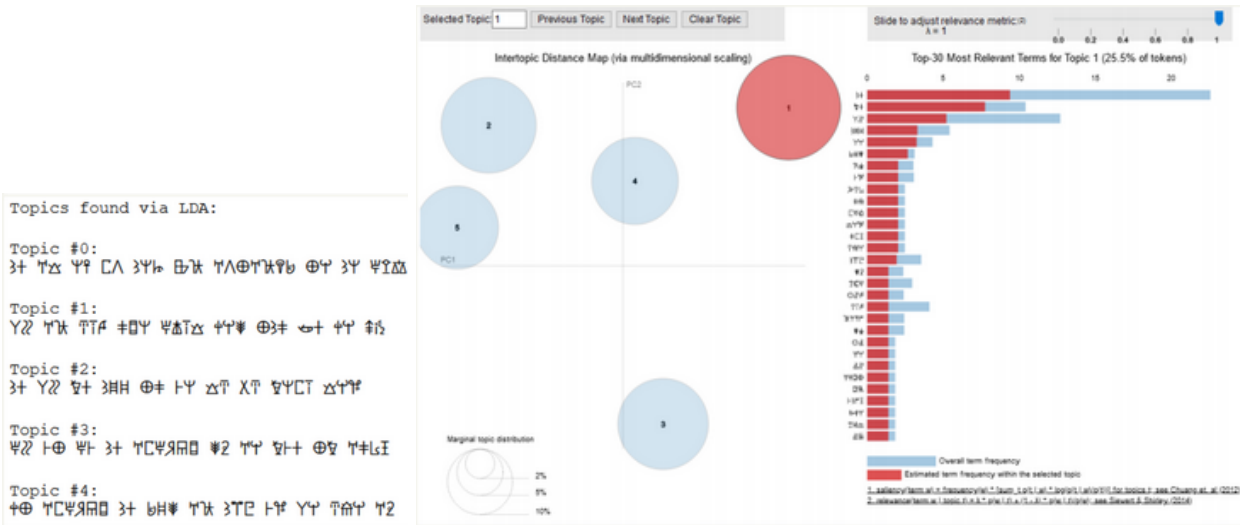


Figure 16: Interactive Topic Modeling graph

pyLDAvis helps to interpret individual topics and their relationships. The circles represent the multidimensional divergence, while the bar chart gives the relevance measure. The bar charts are for both corpus-wide frequency and topic-wise.

The terms are ranked by their relevance in the corpus data with the λ slider. While the terms are ranked by the probability of the topic, since the pyLDAvis is an interactive tool, the topics can be selected and adjusted accordingly. The λ slider adjusts the critical value of the relevant term rank for each specific topic. The value that is usually used is 0.6. The Topic Modeling results go on to show that there are five major groupings of the transcriptions in Linear A language as the Intertopic Distance Map (via multidimensional scaling) shows. While group 2, 3, and 4 are overlapping, which seems to indicate closer similarity, 1 and 5 are on the opposite side of the graph in the modeling graph.

The LDA proved to be an improvement in similarity analysis by topics. While this is one methodology to understand the patterns in the undeciphered language, and the visual gave a better idea of token relationships, by itself LDA was an improvement over the base NLTk entropy function.

2.2.2 Exploratory Knowledge Mining

In addition to NLP, we also used some machine learning methods to detect most common symbols. In order to use machine learning, the data needed to be pre-processed so that it could work best within the constraints of the two following methods we tried: Naïve Bayes[15] (NB) and K-nearest-neighbor[16] (kNN).

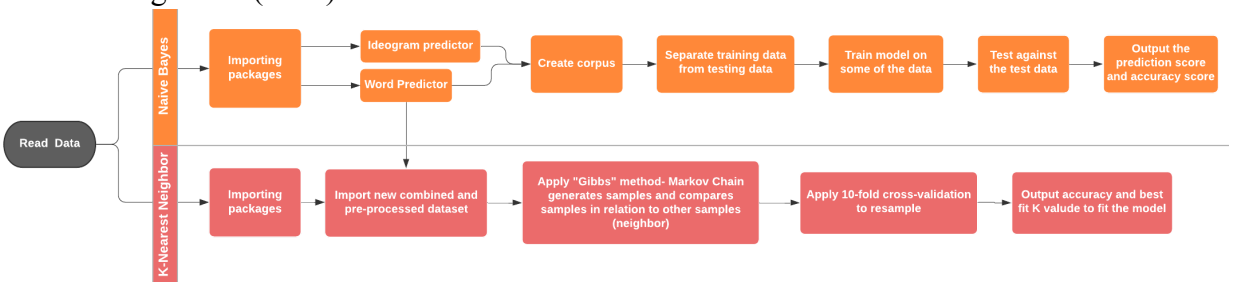


Figure 17: Knowledge Mining process flow

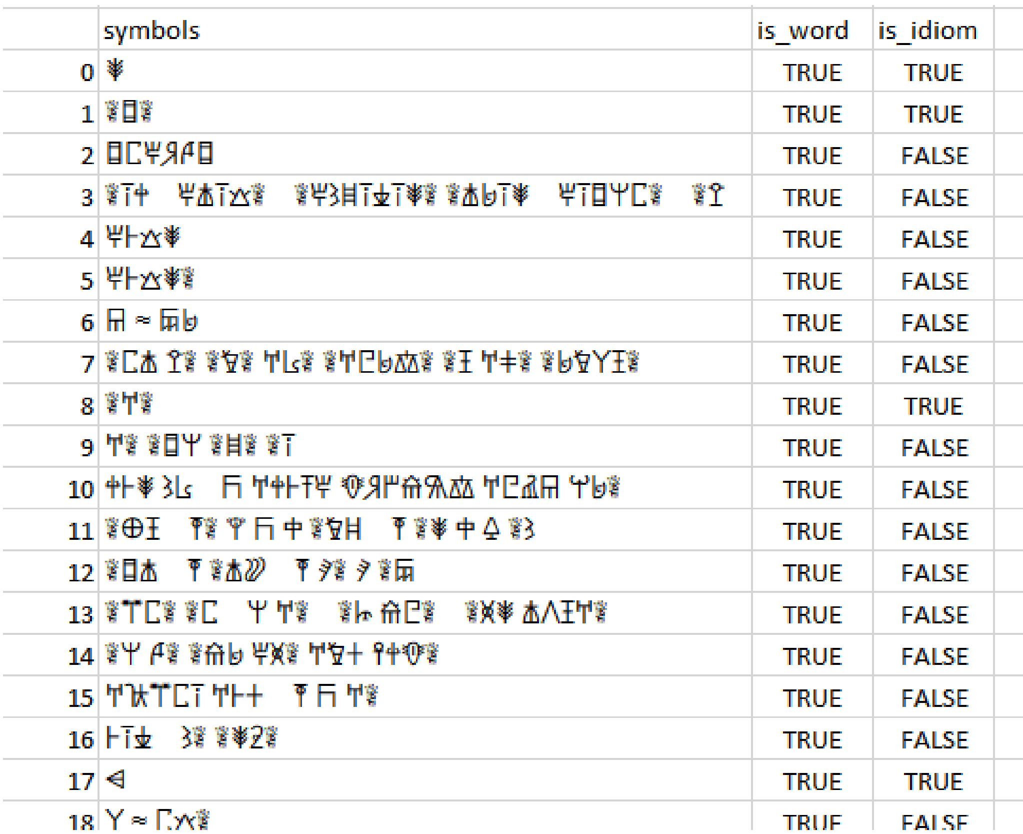
NB uses a probabilistic method for breaking down the components of the symbols. From a data pre-processing perspective, we needed to understand which symbols and representation to use (Bunescu and Mooney, 2004). We opted for using the symbols which were likely words, as well as symbols that were likely ideograms. The difference between symbols and words were great enough so that assigning a true or false to whether the symbol represented a word or ideograms, could allow for NB to make a determination on future symbols or ideograms, or if a symbol fell outside of these categories.

[15]In statistics, Naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong independence assumptions between the features.

[16] In instance-based classification, each new instance (of symbol, in our case) is compared with existing ones using a distance metric, and the closest existing instance is used to assign the class to the new one. This is called the nearest-neighbor classification method. (Witten et al. 2017)

Figure 18: Determination of symbols commonality by their representation

In order to accomplish this, we ran NB on the "word" symbols, as well as the "ideograms" symbols. We used different percentages for training versus testing data, with the results below using 1/3 of the data for training and the remaining data for testing. The accuracy of NB for word detection was 73% with a low precision rate of around 0.45, while the accuracy for ideogram detection was lower with 65% accuracy with a higher accuracy of 0.71. In order to read the symbols, we had to make an assumption on what consisted of a word. Each instance of a symbol represented a word, and ideograms were found in words, which may explain the lower accuracy for ideograms detection. Another point to note is that the dataset included fewer ideograms than words.



	symbols	is_word	is_idiom
0	[illegible]	TRUE	TRUE
1	[illegible]	TRUE	TRUE
2	[illegible]	TRUE	FALSE
3	[illegible]	TRUE	FALSE
4	[illegible]	TRUE	FALSE
5	[illegible]	TRUE	FALSE
6	[illegible]	TRUE	FALSE
7	[illegible]	TRUE	FALSE
8	[illegible]	TRUE	TRUE
9	[illegible]	TRUE	FALSE
10	[illegible]	TRUE	FALSE
11	[illegible]	TRUE	FALSE
12	[illegible]	TRUE	FALSE
13	[illegible]	TRUE	FALSE
14	[illegible]	TRUE	FALSE
15	[illegible]	TRUE	FALSE
16	[illegible]	TRUE	FALSE
17	[illegible]	TRUE	TRUE
18	[illegible]	TRUE	FALSE

Figure 19: Restructuring data to identify probable words as a word or a logographic idiogram

Another machine learning application we used was kNN. In order to use kNN, we needed to understand what we wanted to use as a measure of nearness. The "Gibbs" method, which utilizes a Markov chain in order to establish the next nearest point by generating a chain of samples, seemed like the most flexible use of kNN for a dataset that had not been deciphered, and therefore did not allow decision based on the knowledge of the language. As the "Gibbs" method has been used for words in sentences, the use for this dataset seemed to work theoretically (Malouf, 2002). While "Gibbs" can provide a helpful way to determine nearest when ambiguity

of the text is present, it does so with greater error. With the “Gibbs” method, along with 10-fold cross validation, the accuracy of prediction was around 64%, with a kNN of 7 being used for the final model.

III Conclusion

3.1 Discussion

The NLP analysis gave us the frequency distribution of popular symbols in the Linear A stone tablets. n-gram summary of the tablets’ contents and predicted the possible subsequent symbols in other tablets. With the inclusion of symbols beyond probable words, the entropy reduced from 9.4 to 3 thereby verifying the consistent unstructured nature of the language via word embedding process using Word2Vec and Principal Component Analysis. Three methods of grouping the symbols were explored from Cosine Similarity Analysis, K-Means Clustering, Topic Modeling, and K-Nearest Neighbor verified by 10-fold Cross Validation, all of which were successful in various measures. Majority of the symbols were grouped in similar buckets except a few interesting outliers. Attempting to classify the symbols was an interesting exercise for ascertaining the accuracy of ideograms particularly.

Deciphering Linear A, the ancient Mycenaean language has been an ongoing effort for many decades now. Through our exploratory computational analysis, we hope to add to the discourse via introducing various new methodologies to this study of this language. To the best of our knowledge, this is the first study to discuss and show computational analysis of Linear A. The purpose of the current study was to examine and apply the possibilities of a series of Natural Language Processing and Machine Learning techniques to undeciphered languages. The paper also proposes to broaden the toolkit for these languages and bridging the gap between a linguistic approach and a data-mining approach. Utilizing computational techniques on the undeciphered scripts in Linear A, we yield some preliminary results. Our future work includes a comparison and analysis of Linear A with other ancient languages, both deciphered (Linear B) and undeciphered (Rongorongo).

3.2 Acknowledgments

We are incredibly grateful for Dr. William Kennedy, and Dr. Duoduo Liao’s support and guidance in this endeavour. Further specialized inferences will be gleaned from the results by Dr. Frederico Aurora, in conjunction with our team.

References

"Using Zipf’s Law To Improve Neural Language Models", Medium, 2020. [Online]. Available: https://medium.com/@_init_/using-zipfs-law-to-improve-neural-language-models-4c3d66e6d2f6. [Accessed: 18- Oct- 2020].

A. Bouchard-Côté, D. Hall, T. L. Griffiths, and D. Klein, “Automated reconstruction of ancient languages using probabilistic models of sound change,” Proc Natl Acad Sci USA, vol. 110, no. 11, p. 4224, Mar. 2013, doi: 10.1073/pnas.1204678110.

Aurora, F. (2015). DĀMOS (Database of Mycenaean at Oslo). Annotating a Fragmentarily Attested Language. Procedia - Social and Behavioral Sciences, 198, 21-31. Doi:10.1016/j.sbspro.2015.07.415

Buitinck et al., (2013). API design for machine learning software: experiences from the scikit-learn project.

Bunescu, R. and Mooney, R.J. (2004). Collective information extraction with relational Markov networks. In Proceedings of the 42nd ACL, pages 439–446.

C. E. Shannon., A mathematical theory of communication,Bell System Tech. J., 27 (July 1948), 379–423; (October 1948), 623–656. Reprinted in:C.E. SHANNON, W. WEAVER,The Mathematical Theory Of Communication, The University Of Illinois Press, Urbana, 1949.

Classen, M., & Safrany, L. (1975). Endoscopic papillotomy and removal of gall stones. *British Medical Journal*, *4*(5993), 371–374. https://doi.org/10.1136/bmj.4.5993.371

Daggumati, S., & Revesz, P. Z. (2019). Data mining ancient scripts to investigate their relationships and origins. Proceedings of the 23rd International Database Applications & Engineering Symposium, 1–10. https://doi.org/10.1145/3331076.3331116

E. Matricciani. Shannon's entropy as a measure of the “life” of the literature of a discipline. Scientometrics 30, 129–145 (1994). https://doi.org/10.1007/BF02017218

Emerging Technology. “Machine learning has been used to automatically translate long-lost languages,” MIT Technology Review. https://www.technologyreview.com/2019/07/01/65601/machine-learning-has-been-used-to-automatically-translate-long-lost-languages/ (accessed Oct. 18, 2020).

Evans, A. (1909). *Scripta Minoa: The written documents of Minoan Crete, with special reference to the archives of Knossos*. Oxford ; Clarendon Press. http://archive.org/details/scriptaminoawrit01evanuoft

F.R.S, K. P. (1901). LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *2*(11), 559–572. https://doi.org/10.1080/14786440109462720

Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020). DOI: 0.1038/s41586-020-2649-2. (Publisher link).

J. D. Hunter, "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.

J. Luo, Y. Cao, and R. Barzilay, “Neural Decipherment via Minimum-Cost Flow: from Ugaritic to Linear B,” arXiv:1906.06718 [cs], Jun. 2019, Accessed: Oct. 18, 2020. [Online]. Available: http://arxiv.org/abs/1906.06718.

J. Novembre, “Pritchard, Stephens, and Donnelly on Population Structure,” Genetics, vol. 204, no. 2, pp. 391–393, Oct. 2016, doi: 10.1534/genetics.116.195164.

Jurafsky, D., & Martin, J. H. (2008). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition. Pearson Education.

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, *28*(2), 129–137. https://doi.org/10.1109/TIT.1982.1056489

Luo, J., Hartmann, F., Santus, E., Cao, Y., & Barzilay, R. (2020). Deciphering Undersegmented Ancient Scripts Using Phonetic Prior. *ArXiv:2010.11054 [Cs]*. http://arxiv.org/abs/2010.11054

M. Waskom, “mwaskom/seaborn: v0.11.0 (Sepetmber 2020)”. Zenodo, 08-Sep-2020.

MacQueen, J. (1967). *Some methods for classification and analysis of multivariate observations*. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics. https://projecteuclid.org/euclid.bsmsp/1200512992

Malouf, R. (2002) Markov models for language-independent named entity recognition. In Proceedings of the 6th CoNLL, pages 187–190.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. *ArXiv:1310.4546 [Cs, Stat]*. http://arxiv.org/abs/1310.4546

N. M. Dinesh, S. Narasimhan, P. Prashanth, and S. E. P. Pushpa, “Entropy of Tamil Language and Prioritized Coding Algorithm for Encoding of Tamil Letters,” in 2018 International Conference on Smart Systems and Inventive Technology (ICSSIT), Dec. 2018, pp. 395–397, doi: 10.1109/ICSSIT.2018.8748705.

R. Lee, P. Jonathan, and P. Ziman, “Pictish symbols revealed as a written language through application of Shannon entropy,” Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 466, no. 2121, pp. 2545–2560, Sep. 2010, doi: 10.1098/rspa.2010.0041.

Rudolf Clausius. (1867). *The Mechanical Theory of Heat: With Its Applications to the Steam-engine and* ... J. Van Voorst. http://archive.org/details/mechanicaltheor04claugoog

S. Bird, E. Loper, E. Klein (2009), Natural Language Processing with Python. O’Reilly Media Inc.

Sievert, Carson & Shirley, Kenneth. (2014). LDAvis: A method for visualizing and interpreting topics. 10.13140/2.1.1394.3043.

T. Petrolito, R. Petrolito, F. Perono Cacciafoco, and G. Winterstein, “Minoan linguistic resources: The Linear A Digital Corpus,” in Proceedings of the 9th SIGHUM Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities (LaTeCH), Beijing, China, Jul. 2015, pp. 95–104, doi: 10.18653/v1/W15-3715.

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2017). Data mining: Practical machine learning tools and techniques.

Zipf, G. K. (1965). *The psycho-biology of language: An introduction to dynamic philology*. Cambridge, Mass. : M. I. T. Press. http://archive.org/details/psychobiologyofl0000zipf

“mwenge/lineara.xyz,” GitHub. https://github.com/mwenge/lineara.xyz (accessed Oct. 18, 2020).

ANNEX 1

Appendix

https://github.com/mwenge/LinearA

Glossary

Python packages

NLTK: Natural Language Toolkit

A set of Natural Language Processing algorithms written in Python.

seaborn: statistical data visualization

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

Matplotlib: Visualization with Python

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations.

NumPy: Scientific Programming

The fundamental package for scientific computing with Python

Scikit-learn: Machine Learning in Python

Simple and efficient tools for predictive data analysis

wordcloud : word cloud generator in Python.

Pandas: Data analysis and manipulation tool

pyLDAvis

Python library for interactive topic model visualization

Extracting and organizing the Linear A Text

The complete Linear A corpus is relatively small, consisting of only a few hundred documents. The total corpus would fill only 5 or 6 pages if all the documents were placed end to end. A Linear A scholar George Douros has produced a spreadsheet that tabulates each document and helpfully provides word breaks, ideograms, and numerals broken out for each line of each document. When converted to csv format this spreadsheet allows me to create the core of the app, which is a structured representation of each document in JSON format. Github user mwenge has a python script that takes in the csv file and writes a javascript array of JSON objects to a javascript file called LinearAInscriptions.js.. Here is a snippet of the file, showing how the document HT1 is represented in as a JSON object:


```
["HT1",{
    "image": "images/HT1-Inscription.jpg",
    "name": "HT1",
    "parsedInscription": "□□□.\n□□◦⁝⁝⁝¦¦¦\n□□≣≡\n□□□□≡="\n□□□◦⁞⁞\n□□□□◦¦¦",
    "tracingImage": "images/HT1-Tracing.jpg",
    "transcription": "□□□.□□◦⁝⁝⁝\n¦¦¦□□≣≡□□\n□□≡="□□□◦\n⁞⁞□□□□◦¦¦",
    "translatedWords": [
    "QE-RA2-U",
    ".",
    "\n",
    "KI-RO(owed)",
    "197",
    "\n",
    "□-SU",
    "70",
    "\n",
    "DI-DI-ZA-KE",
    "52",
    "\n",
    "KU□-NU",
    "109",
    "\n",
    "A-RA-NA-RE",
    "105"
    ],
    "transliteratedWords": [
```

```
                "QE-RA2-U",
                "·",
                "\n",
                "KI-RO",
                "197",
                "\n",
                "□-SU",
                "70",
                "\n",
                "DI-DI-ZA-KE",
                "52",
                "\n",
                "KU□-NU",
                "109",
                "\n",
                "A-RA-NA-RE",
                "105"
                ],
                "words": [
                "□□□",
                "·",
                "\n",
                "□□",
                "𐄀𐄁𐄂𐄂",
                "\n",
                "□□",
                "𐄁𐄂",
                "\n",
                "□□□□",
                "𐄂𐄂",
                "\n",
                "□□□",
                "𐄀𐄃",
                "\n",
                "□□□□",
                "𐄀𐄂"
                ]
}],
```


The inscription that we need to visualize is encoded in the code above. The image files, the raw transcription, and arrays representing the parsed words of the inscription both in Linear A, transliterated Linear A syllabograms, and proposed translations where applicable are included in this.

ANNEX 2

Cras tristique vel nisi at aliquet. Proin egestas erat sit amet velit lobortis imperdiet. Integer et arcu sapien. Etiam id blandit sapien. Nam tempus lacus ac massa semper, vel laoreet turpis rutrum. Mauris eget nibh vitae justo porta imperdiet sed vel ligula. In imperdiet, augue vel condimentum convallis, neque augue imperdiet neque, eget dapibus nunc mauris ultricies tortor. Nam eget nunc egestas, blandit lectus non, aliquam nunc. Cras sed quam vitae arcu ornare lobortis. Ut ut lacus hendrerit, convallis orci sit amet, commodo nunc. Pellentesque eget tincidunt tortor. Nunc ornare molestie mauris id vehicula. Suspendisse pharetra tortor metus, sit amet fermentum tellus vehicula ut.