Who Wrote It?

A Machine Learning Analysis of Letters from Mari Diplomats

Mateo Abascal

1 Introduction and Background

1.1 The Debate

This analysis begins with an open question: who actually wrote the texts sent by Mari diplomats and generals stationed outside Mari? According to Jack Sasson, "written statements were read aloud by scribes to illiterate officials,"[1] so presumably the "illiterate officials" also dictated their letters to the scribes. This, of course, raises the question of whether entire letters were dictated word for word, or perhaps just the key points were given to the scribes who then wrote the letters themselves. Additionally, where were these scribes from? Would a Mari diplomat stationed in Babylon use a Babylonian scribe or bring a scribe with him in his envoy? But then Dominique Charpin decided to ask if scribes were even involved at all. In his book *Lire et écrire en Mésopotamie*, he makes the claim that, not only were the "illiterate officials" actually literate, they also may have written some of their own documents (though perhaps not letters).[2]

Charpin rests his argument almost entirely on a few examples of generals or diplomats who were clearly able to read. This doesn't prove they were capable of writing though, nor does it guarantee they would have chosen to write their own letters given they could have easily hired a scribe. One text he cites from Zimri-Lim to a general of his,

[1] Sasson, 1995
[2] Charpin, 2010

Yasim-El, specifically says "these texts, read them yourself,"[3] suggesting that, while Yasim-El was probably literate, he had enough of a tradition of someone else reading his mail to him that Zimri-Lim thought that was a necessary preface. Charpin doesn't really seem to be making a claim that Mari officials wrote all their own letters and hated scribes or anything outlandish. His argument is more focused on the literacy of the generals / officials, what they did with that literacy was up to them.

1.2 The Purpose of this Analysis

Frankly, the purpose of this analysis wasn't to prove a hypothesis or make any claim; the main goal was to give more evidence and context to the claims made by Charpin and Sasson, hopefully shedding light on the lives of Mari diplomats. Charpin and Sasson are incredibly distinguished scholars, but they each stake their claim on a relatively small set of data. In theory, if someone could go through and match up these texts to a model for a canonical Mari text and a canonical non-Mari text, they might shed some light on the debate, but to do so by hand would take far longer than its worth. A neural network, however, could make this a much easier task.

2 Procedure

2.1 A Computational Approach

The first step in clarifying who wrote a text is figuring out the regional dialect in which it's written. This can be accomplished by matching the dialectical trends specific to a certain region. The Mari shift is a simple example of this. Since the Mari shift is specific to certain regions further north from Babylon, the presence of a Mari shift in a text would

[3] Charpin, 2010

suggest that its author is from Mari or perhaps further north, but likely not Babylon. Though language is never perfect, so simply finding a Mari shift in a text doesn't guarantee the text wasn't written by a Babylonian scribe.

The approach here is to train a neural network to recognize a Mari shift, and then, using the presence of Mari shifts in letters from Babylonian officials in Babylon and Mari officials in Mari (or the surrounding regions where Mari shifts are present), train a model representative of a Babylonian scribe and one representative of a Mari scribe. This network can then, when given a new text, determine which class it most likely belongs to.

2.2 Selection of Texts

The texts used in this analysis were chosen to represent a variety of officials from each group. Since the Mari shift is not specific just to Mari but to the greater region, some of Mari's vassals have been included in this analysis as well. A complete bibliography of texts is included, but the transliterations are accessible on Github to keep this paper from being too cluttered. Due to the sheer volume of texts, transliterations were pulled from Archibab.fr, and these were later formatted to fit the needs of the neural network.

3 Chosen Corpus

3.1 Babylonian Officials in Babylon

This section is largely pulled from *Altbabylonische Briefe* Volume 4. Specifically texts numbered 1-42 which are all correspondences between Hammurapi and Šamaš-Hazir. Hammurapi wrote in quite a few different media, but diplomatic letters were chosen here since that is the main source material for the Mari officials in Babylon, and this removes as much bias as possible.

3.2 Mari Officials in Mari

This section contains a combination of texts from Zimri-Lim and Ibal-Addu. Letters from Ibal-Addu to Zimri-Lim make up the vast majority of the corpus since, being a king under Zimri-Lim, Ibal-Addu likely would have written to him in a similar form to the diplomats in Babylon. These texts come exclusively from ARM 28.

3.3 Mari Officials in Babylon

This section is influenced by Raymond Cohen's chapter in Jan Melissen's book *Innovation in Diplomatic Practice*. He gives an outline of four Mari officials who served over a year each in the Court of Hammurapi. Letters written back to Zimri-Lim have been sourced from three of these four, Yasim-Hammu, Yarim-Addu, and Šarrum-Andulli, and they are the primary texts this paper seeks to analyze.[4] Each text is an account to Zimri-Lim as they were stationed in Babylon while under his reign. These texts may have been written by the men themselves, perhaps by scribes, but the primary question is whether any Babylonian scribes were involved in these correspondences. The letters themselves are all included in *Archives Royales de Mari*, specifically those of Yarim-Addu are ARM 26/2 361-371, Šarrum-Andulli is responsible for ARM 26/2 376-381, and lastly Yasim-Hammu has provided one letter to this analysis, ARM 26/2 383. There are 4 more letters likely from Yarim-Addu included in ARM 26/2, they are texts numbers 372-375. They have not been included due to unreliability.

[4] Cohen, 1999

4 Computational Theory

4.1 Formatting

Computers and human speech were never truly made to work together. The simple complexity of parsing strings and Text to Speech programs make this pretty clear. Though human speech may not even be as difficult as notations designed specifically for humans to read, such as transliterated Akkadian. Unfortunately, that means that formatting the transliterated texts was possibly the most time consuming part of this process.

The texts were accessed from Archibab,[5] where the provided transliteration was copied to a text file. Each text was then parsed, separating the words and storing them as strings. These were then aggregated into larger arrays separated by line. The lines were then aggregated into a multi-tiered array for each text. Lastly, these were all combined into a JSON dictionary containing, for each text, its name, origin (simplified to Mariote or Babylon), the official who sent it, the official's nationality (also simplified to Mariote or Babylonian), and the formatted transliteration.

It's probably worth noting here why JSON was chosen over, say, ORACC's library of tools, which are specifically designed for cuneiform. The short answer is simplicity. This analysis is very computation heavy, so fully lemmatizing the entire text would not just be overkill for such an analysis, it would also slow the process down. JSON is incredibly lightweight and compatible with just about every programming language, which also means if anyone else would like to further this analysis or contribute to the corpus, they can do so easily via the associated Github repository.

[5] archibab.fr

4.2 Why a Neural Network?

A neural net, unlike most computational algorithms (finite automata), is not programmed step by step. Possibly the most fundamental concept in Computer Science, automata are algorithms that receive an input and decide if it is or isn't part of the larger group. A neural network is the exact opposite. It still has plenty of finite code (building one is more linear algebra than computer science), but the code isn't designed to tell the algorithm what to do, it's designed to show the algorithm what to do. It provides the algorithm with labeled sample data to give the algorithm a chance to adapt and begin recognizing the patterns. A neural net, put simply, is between a normal if / then statement and an artificial intelligence. It's still incredibly finite, but it has far more capabilities than a simple if / then linear program, which makes it perfect for this analysis.

People make mistakes and the universe is never constant. Those are probably the two most difficult things to account for with an algorithm. A neural net gets around this by not relying on strict commands and instead relying on its ability to recognize patterns. Will it always be right? Absolutely not. But the purpose of this procedure isn't to correctly interpret every single tablet and instance of the Mari shift, it's to be accurate enough that one or two errors don't significantly change the conclusion. This is where the Bag of Words model shines.

4.3 Training the Models

Bag of Words was developed for early computer vision and document identification[6] and the theory behind it is relatively simple: take all the words, throw them in a bag, and

[6] Harrison, 2019

then count them. Grammar doesn't matter, just multiplicity. This seems absurdly low tech but it's actually how spam emails are filtered to this day. Starting here and identifying the very very broad trends (with enough data this would begin recognizing *um-ma ha-am-mu-ra-bi-ma* and understanding that this can't be a Mari text, even without knowing who Hammurapi was). This is a good start, but it's not very impressive. The model does set the groundwork for the rest of the computation, however, since it establishes the use of vocabularies. Training such a model results in it having a rudimentary Babylonian and Mari vocabulary. This vocabulary is less focused on capturing all the words, but rather the frequency of specific words, and profiling the dialect as such. This process is iterated so the model can recognize the same word spelled multiple different ways (but can also make a note of it in the case the spelling may be unique). To make this more accurate, rather than simply counting the times a term appears, they are weighted based on presence in one corpus and rarity in the other. This is done by, instead of counting every instance of every term as 1, calculating its tf-idf (term frequency -- inverse document frequency) and summing them over the set.[7]

The problem with just looking at words is that there aren't a lot of them. The more data a model has to work with, the more accurate it will be, so why not break the words down to signs? Or why not go even further and break the signs down into uniliteral phonemes? This is actually an essential step to smoothing out the vectorizations resulting from the tf-idf ranking. Tf-idf is certainly progress in the right direction, but it's still too variable, and without enough words, some random unimportant words will rank much

[7] Wu et al, 2008

higher than they should simply out of sheer unimportance. Running the same algorithm on characters helps to emphasize not just the differences in specific word frequency between documents, but also whether any vowel shifts, assimilations, or nasalizations have taken place, and this is how the algorithm builds an understanding of the Mari shift. It would be very difficult to train a neural network to detect a Mari shift head on, but by teaching it to recognize and emphasize the different spellings of the same word or seemingly odd uses of an *e* vowel, it can develop an understanding of when an *e* is the result of Mari shift or if it dates back further to perhaps a 4th aleph dropping out. This code is on Github and the trained models are identifiable as datasets of vectors. A "Count Vector," "Tfidf Vector," and a "Shift Vector" are the three constituent parts of the following analysis.

4.4 The Guessing Game

Once the models are trained, they're ready to be tested. This testing involves randomly selecting 75% of the data to use in creating a new model, and then test the other 25% on this model. Testing is the relatively simple part, but analyzing this data also requires choosing the right procedure, and just because a model is very specific doesn't mean every algorithm here will do a good job of placing the text. Either way, the text needs to be vectorized just like all the others. Fig 1 shows a graph of the weighted tf-idf datasets for Mari and Babylon plotted on top of one another. This will be properly used and interpreted later, but for now it's only important to note that overlapping colors are areas where it is difficult to distinguish the datasets, but areas that are all one color are signs of some trend that the algorithm picked up on. The point was placed there by the algorithm because that same trend is what pulled the other nearby points so far away from the origin.

The natural grouping that this algorithm invokes invites the use of a KNN or K Nearby Neighbors algorithm.[8] This algorithm is usually described by the instructions "you tell me who your friends are, and I'll tell you who you are." The idea is that, after being placed on such a plot, the text checks the labels of its K nearest neighbors, and then guesses that it's the average of these (depending on the way this is calculated and carried out, the average in question might be the mode or the mean). In the cases there is a clear grouping, this is by far the most reliable method of classification, but it's also very slow. When KNN looks too slow or simply too homogenous and not worth the trouble, the text will attempt to classify itself via a Gaussian Naïve Bayes algorithm. This isn't as visually pleasing, but by using Bayesian statistics, and the naïve assumption that probabilities are conditional and they follow a Gaussian distribution, some smaller claims can be made where there isn't enough variability for KNN to be conclusive. This is useful also in calculating it's own and the algorithm's own probability of being correct, in addition to helping classifying the text.

5 A Word of Caution

5.1 Broken and Reconstructed Text

If a letter starts with a broken first line ending in *bi$_2$-ma*, it may not be that dangerous to reconstruct an *[a-na PN qi$_2$]-bi$_2$-ma*. In general, reconstructed text actually shouldn't be a problem as it simply adds more data for the computer to work with, and statistically it shouldn't change the predictions that much either. However, this was a very significant problem for the character recognition side. Words tend to be easier for the network to interpret, as it can make out the intricacies of human speech. In the case of

[8] Harrison, 2019

ARM 26/2 381, there is a break in line T22. The break is specifically one letter: *l[i]-im*. While the network was training it was given a significant amount of pseudo-fake texts. These texts were real texts, but parts were intentionally removed. The theory here was akin to when someone starts getting better and better at Sudoku and the only way to keep getting better is to be given fewer starting numbers. These breaks were fine in a word only algorithm, but when character support was added in, this became a very significant problem, especially, as expected, with the letter *e.*

Plenty of languages from Arabic to ancient Egyptian intentionally abstain from writing short vowels because it simply isn't necessary. Most fluent English speakers, knowing the game here is to recognize the word without vowels, could see the letters *wtr* and figure out that they are probably short for *water*, maybe *waiter*. In context, written *I would like a glass of wtr*, it's pretty clear. As a side anecdote, even the Spell Check used on this paper tried to correct *wtr* to *water*. Similar to how it's relatively easy for a human, it's incredibly easy for a computer to see, assuming it is trained with an unbiased data set. Given a set containing 20 different contexts of the word *waiter* but only one sentence *the waiter poured more water*, a neural net would probably say "*the waiter poured more waiter* is spelled wrong." But this all changes with character analysis.

Character analysis is an incredibly powerful tool to round out the variability of a word based neural net, and also to detect spelling changes across corpora, but it should never be given too much weight since it's completely incapable of making substantial conclusions on its own. This neural net fell very quickly into the trap of realizing that it was very often correct if it said texts with more *e*'s were from Mari and less *e*'s were from

Babylon. AbB 4 19, to this point, has 9 *e*'s for only 22 lines, which is way higher than most Babylonian texts; it's obviously from Mari! But alas line 3 reads *um-ma ha-am-mu-ra-bi-ma*; the neural net was wrong. One or two *e*'s makes all the difference, and so the neural net would get repeatedly confused when introduced to (pseudo-fake) reconstructed *e*'s.

The solution is relatively simple: get rid of reconstructions. Character recognition, and it's ability to detect the Mari shift, is far more important than the little bit of extra data gained from a few additional signs. Of course, the point of a neural network is that, while it can be trained on an ideal dataset that may not exist in the real world, it should work on any dataset, even if that dataset is full of reconstructions. The middle ground, then, is to allow reconstructions for word based analysis but not for character analysis. Since the two algorithms don't involve one another until they have each vectorized the text, it doesn't matter if their data is ever so slightly different

5.2 Mari Shift "Imposters"

The other significant problem with a neural net looking at the Mari shift is that, on the surface, there are a lot of things that look like the Mari shift. With a dictionary on hand or some cursory knowledge of Akkadian vocabulary, this isn't very hard to identify, but solely via pattern recognition, its near impossible to do confidently. The neural net would need to see a lot of the exact same words written without the shift to recognize it with the shift. While the algorithm is fully capable of getting better, it's terrible at first. For starters, it double or sometimes even triple counts Babylonian vowel harmony on the character recognition side, so it's effectively useless at directly identifying it. In theory it should

actually do better at identifying a lack of the Mari shift. This is solved by training the model many times, and properly weighting the value of the character analysis vectors, making sure that the significant findings regarding a possible Mari shift are recorded when the neural net believes it has seen a similar word in Babylonian without the *e*, or vice versa for a lack of Mari shift when it is allowed.

6 Findings

6.1 Final Analysis

For the sake of brevity, this section will not cover the computations made to arrive at these statistics. The code is available as an open source project and substantial graphs have been provided in the Appendix which will be referenced here.

Fig 1, which will now be discussed in depth, is a plot of all the vectorized, weighted scores from the Babylonian corpus overlayed on the vectorized, weighted scores of the Mari corpus. Mari texts are in blue, and Babylonian texts are in red. The clustering algorithm boasts a 98% accuracy at separating these. Logically, there is quite a bit of overlap around the origin of the plot: some texts just weren't particularly Babylonian or Mariote. The extrema are really the focus. There is a long very thin Babylonian spike in the top right corner. This lines up with the fact that the Babylonian dataset used was a bit slim and homogenous, and one person's name, ^{d}*UTU-ha-zi-ir*, was repeated many times. This spike, as well as the little cloud centered around (0.3, 0.1) just so happen to be related to this. Shorter texts and repeated names not present in the other corpus make for very damaging outliers. The important part here is the blue flower petal from the origin to (-0.2, 0.4). Short of further investigation and more data, this seems to be the result of the Mari

shift. The texts appearing here seem to, upon a cursory analysis, have more instances of Mari shifts, though clarifying exactly what lies behind this petal is might be an interesting subject for further research. Whatever the petal represents, however, is present in the texts from the Mari diplomats in Babylon as well, as Fig 2 illustrates.

6.2 Interpretations Regarding Mari Diplomats

Figure 2 is a plot of all the vectorized, weighted scores from the Diplomatic corpus overlayed on the vectorized, weighted scores of the Mari corpus. Again, Mari texts are blue, and this time the Diplomatic texts are green. As was the case with Fig 1, there is a pretty unusable overlap around the origin. Notably, however, the Diplomatic vectors fall almost perfectly along the petal. This is emphasized in Fig 3, which is the same plot but layering back in the Babylonian texts in red, where there is a very clear overlap between Mari and the Diplomats, and a very clear lack of Babylonian texts along the petal. All of this comes with an analysis of 1240 data points, and as of the last time it was run, only 8 were mislabeled, signifying over 99% confidence that the people writing these correspondences were Mariote (So far 18 is the most it's ever gotten wrong, which is still 98.5%). I leave it open to Charpin to decide if it was the officials themselves or scribes they brought from home, but due to the fact that even Charpin doubted whether officials would write their own letters, I believe it is relatively safe to say that Mari diplomats brought their own scribes with them (at least to Babylon), and that this neural network is capable of identifying the characteristic features of the Mari dialect of Akkadian (insofar as they differ from Babylonian Akkadian).

6.3 Future Research

This analysis, while robust enough to make the aforementioned conclusion, could have been far more complex. For one, the use of a nonlinear dimensional reduction algorithm would result in much clearer boundaries on the plot. Additionally, this model was designed to be monitored, whereas a fully independent model could learn much more over time. I would also like to investigate the specific characteristics of the plot, given that, admittedly, PCA is messy enough and collapses and rebuilds enough data that it's difficult (though not impossible) to confidently identify any region of the plot without attempting to remove specific texts and re run the entire network. The similarities and differences within the plot are still very valuable for making a conclusion, but the cloud at (0.3, 0.1) for example might be the result of another trend that wasn't accounted for in the selection of these texts, and perhaps a reoriented KNN clustering algorithm would see the cloud not as overlap of 3 corpora but a corpus of its own.

While a neural network may not be able to understand as much as a professional in the field, it is an invaluable tool to help recognize patterns that most people would have missed. Who knows where this may be applicable next.

References

"1.9. Naive Bayes." scikit. Accessed December 14, 2019.

Abascal, Mateo. "Mateo/MariShift." GitHub, December 14, 2019.

Charpin, Dominique, and Jane Marie. Todd. *Writing, Law, and Kingship in Old Babylonian Mesopotamia*. Chicago: University of Chicago Press, 2010.

Cohen, Raymond. "Chapter 1: Reflections on the New Global Diplomacy: Statecraft 2500BC to 200n.d.." In *Innovation in Diplomatic Practice*, 1–21. New York City, New York: Palgrave, 1999.

Harrison, Onel. "Machine Learning Basics with the K-Nearest Neighbors Algorithm." Medium. Towards Data Science, July 14, 2019.

Heimpel, Wolfgang. *Letters to the King of Mari A New Translation, with Historical Introduction, Notes, and Commentary*. Winona Lake: Eisenbrauns, 2014.

Pedregosa, Fabian, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, and Bertrand Thirion. "Scikit-Learn: Machine Learning in Python." *Journal of Machine Learning Research* 12, no. 2825-2830 (November 10, 2011).

Wu, Ho Chung, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. "Interpreting TF-IDF Term Weights as Making Relevance Decisions." *ACM Transactions on Information Systems* 26, no. 3 (January 2008): 1–37.

Appendix

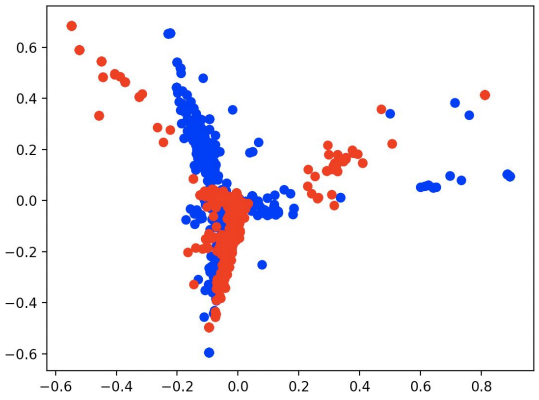

Figure 1

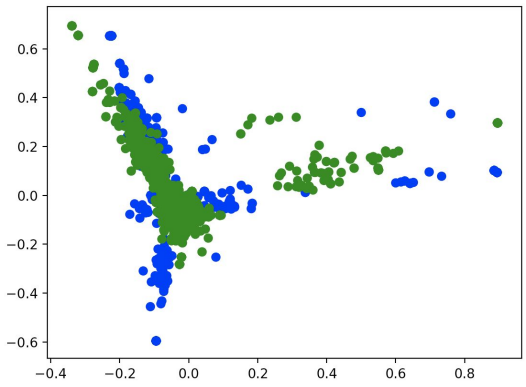

Figure 2

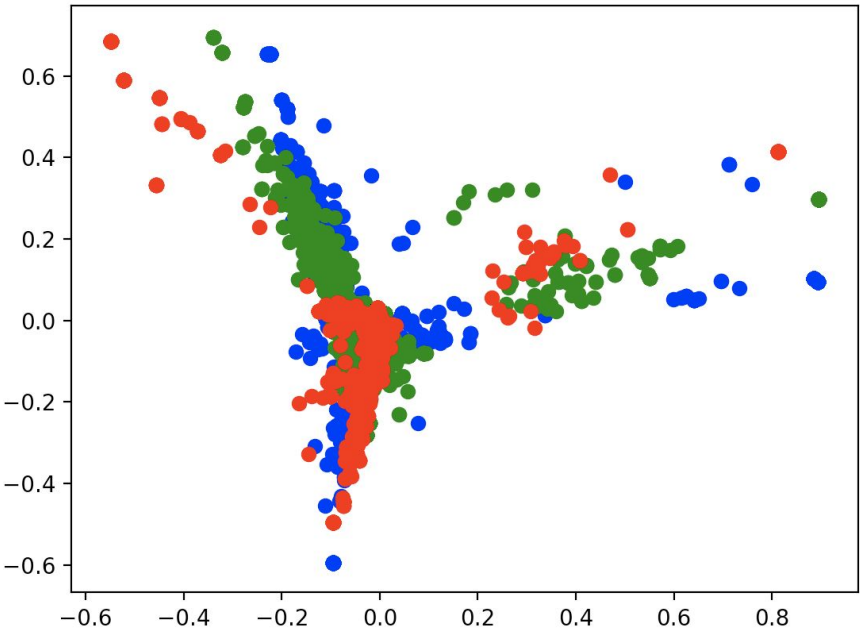

Figure 3

List of Texts

Babylonian Officials in Babylon

- ARM 28 1 / Hammurapi → Zimri-Lim
- ARM 28 6-7 / Hammurapi→ Buqāqum
- ARM 28 8 / Hammurapi → Bahdi-Lim
- AbB 4 1-42 / Hammurapi → Šamaš-Hazir

Mari Officials in Mari

- ARM 28 2, 11-13 / Zimri-Lim → Hammurapi
- ARM 28 15 / Zimri-Lim → Amut-Pi-El
- ARM 28 16 / Zimri-Lim → Yarim-Lim
- ARM 28 48-74, 77-78 / Ibal-Addu → Zimri-Lim
- ARM 28 75 / Ibal-Addu → Šunuhra-halu

Mari Officials in Babylon

- ARM 26/2 361-371 / Yarim-Addu → Zimri-Lim
- ARM 26/2 376-381 / Šarrum-Andulli → Zimri-Lim
- ARM 26/2 383 / Yasim-Hammu → Zimri-Lim