Voynich2Vec:
Using FastText Word Embeddings for Voynich Decipherment

William Merrill
Yale University
william.merrill@yale.edu

Eli Baum
Yale University
eli.baum@yale.edu

1 Introduction

1.1 Word Embeddings

[wm] Word embeddings, or vector representations of words, are a useful way of representing lexical semantics for a variety of natural language processing (NLP) applications. The idea behind this approach is to encode each word as a high-dimensional vector of real numbers such that similar words have similar vector representations. The similarity between two word vectors is often assessed in terms of cosine similarity, which is defined as

$$\text{cos-sim}(w_1, w_2) = \frac{w_1 \cdot w_2}{\|w_1\|\|w_2\|} \quad (1)$$

and gives the angle between the vectors. Vectors that point in the same direction will have a cosine similarity of 1, and vectors that are perpendicular will have a cosine similarity of 0. In the case of word embeddings, word vectors for highly similar words have a cosine similarity close to 1, and words that have very little to do with one another will have a similarity score close to 0.

Most modern approaches to building word embeddings trace from the word2vec method developed by Mikolov et al. (2013), which used a neural network to learn from raw text an embedding that encodes the distributional context in which a word can appear. A neural network is a sophisticated machine learning algorithm that has achieved notable success in NLP and other problem domains during the last decade. Importantly, since a word2vec network only needs raw text to learn embeddings, it can be run on the Voynich manuscript. This is possible because the algorithm assumes that the context in which a word appears encodes that word's meaning, which is justified empirically by the success of Mikolov et al. (2013)'s method and theoretically by the Firthian tradition in linguistics (Firth, 1957).

Lample et al. (2017) developed a modification of the core word2vec architecture called fastText that was designed to incorporate the morphological properties of words into its learned notion of distributional similarity. Rather than directly learning embeddings for words, fastText learns embeddings for the character n-grams appearing within words. Then, a word embedding is produced by combining the n-gram embeddings for all the n-grams in a word. From a linguistic point of view, this is very similar to asking which sequences carry morphosyntactic information. The size of n-grams that should be considered is a hyperparameter of the model that needs to be set. The end result is a word embedding scheme whose notion of distributional context incorporates more morphosyntactic information than word2vec.

1.2 Word Embeddings and Voynich

Perone (2016) reported some preliminary analysis of the Voynich manuscript using word2vec embeddings. Using an established dimensionality reduction technique called t-distributed Stochastic Neighbor Embedding (t-SNE), Perone transformed these word embeddings into two-dimensional space in an effort to visualize them. The plots that he reported had two primary clusters which he claimed corresponded to the two different hands in the manuscript. Based on this, Perone tentatively suggested that the hands might be two different languages (or at least two different modes of text generation).

Perone (2016) also analyzed the embedding of one star label in an astrological diagram, noting how the embeddings clustered around it also seem to be star names. This sort of bottom-up analysis might suggest that there is some semantic content in the manuscript that word2vec can pick up on. However, Perone only presented analysis of this

one word. Thus, there is more potential for work in this vein.

Instead of using word embeddings, Bunn (2017) built embeddings for each folio in the manuscript using the frequency of each character. Each page was represented by a vector encoding the relative probability of occurrence of each character on that page (with a 24-character EVA alphabet, each vector had dimension 24). Bunn (2017) then used the same t-SNE visualization to plot the folio vectors in three-dimensional space. His intention was to use this data to validate Currier's Language A/B hypothesis (which was primarily made on the basis on character probabilities). Bunn's results show that folios tagged as A tend to appear in a different part of the vector space than those tagged as B, although there is some overlap between the regions. Since the boundary between Languages A and B is poorly defined, it is unclear whether many of the folios belong to a cluster.

1.3 Word Embedding Alignment

Unsupervised techniques for aligning fastText embeddings could also prove useful for trying to make sense of the Voynich lexicon. Word alignment is the task of matching up words in different languages with the same meaning. Lample et al. (2017) developed an approach to this problem called Multilingual Unsupervised or Supervised Word Embeddings (MUSE), which uses an adversarial neural network to align word embeddings from different languages without parallel texts. The intuition behind the learning algorithm is that one part of the neural network tries to transform vectors in one language into the other, and the other part of the network tries to distinguish between the transformed vectors and actual vectors in the other language. These adversarial training dynamics can eventually lead to a neural network that is able to map vectors from one language to the other. Given large data sets, the model achieved close to state-of-the-art performance. While this approach could in principle be used to align Voynichese words to, say, Latin words, the amount of Voynichese text in the manuscript is much smaller than the massive corpora typically used to train such sophisticated models. This lack of data is confounded by the fact that the model assumes there is a linear transformation between the semantic spaces of each language, which is problematic if the topics of two small documents are dramatically different.

We aimed to build on the work done by Perone (2016) by training fastText embeddings on the text of the Voynich manuscript. We will analyze these embeddings to address the following three questions:

1. **Morphosyntactic Clustering** Do the fastText vectors reveal any suffixes that might function as morphological units?

2. **Topic Modeling** Do we find similar vector representations for folios with similar illustrations?

3. **Unsupervised Word Alignment** Can we decipher any Voynich words by aligning their word embeddings with embeddings from other languages?

2 Data

[eb] We built our word embeddings from the 1999 Takahashi transcription of the manuscript. While the archive file (`text16e6.evt`) is nearly machine-readable, it needed to be cleaned up slightly.

The first step in our investigation was to write a "tokenizer" which read the transcript file and produced a list of Takahashi EVA words. The main difficulty here was accounting for special symbols that had been inserted into the transcription to denote illustrations and formatting. For example, if a flower interrupts a word, `{plant}` would be inserted in the middle of a word. Elsewhere, extra spaces had been inserted to align the Takahashi transcription with other transcriptions, or show where folds and tears in the parchment were. Manual analysis and removal of such tags allowed us to create a reliable and usable transcription.

While creating the tokenizer, we did notice some errors and ambiguities in our source transcription. While many of these ambiguities are caused by smudged ink or torn pages, a number are very clear on the Beinecke's recent scans. One direction for further work is to create an updated transcription using these new high-definition images. However, for our project, we chose not to modify Takahashi's transcript from the original, to allow for a (hopefully) more consistent analysis.

3 Methods

3.1 FastText

[wm] We built our fastText vectors using a public Python implementation (Yansyah, 2017-). Following the default values in the software package, we chose to train a skipgram model using n-grams of size 3 – 6. In Section 4.1, we also investigated the effects of switching to n-grams of size 2 – 6 when we were analyzing length 2 suffixes. Also based on the default parameters of the software package, we excluded words that occurred less than 5 times in the manuscript.

Recall that fastText embeddings encode some representation of the morphosyntactic distribution of their corresponding words. This means that, if a suffix encodes morphological information, some of the fastText embeddings for words ending with that suffix should cluster. This should hold especially when we don't have a large enough data set to determine meaningful semantic information about all roots.

Looking at word embeddings learned from *Secreta secretorum*, a Latin manuscript of similar size to Voynich, we verified that embeddings did in fact show clustering by functional morpheme. For example, one of the embedding pairs with the highest cosine similarity was *pulcritudine* ("beauty-ABL") and *longitudine* ("longitude-ABL"). These two words share both the derivational nominalizing suffix *-tud-* and the ablative case marker *-e*, which explains why they should have very similar morphosyntactic distribution. It is worth noting that words of higher frequency tended to cluster by root as opposed to functional suffix. For example, the closest alignment overall was *Alexandri* ("Alexander-GEN") with *Alexandro* ("Alexander-DAT/ABL).

Inspired by these observations, we developed a method to test the morphosyntactic contribution of suffixes in Voynich. We chose several commonly occurring suffixes in the manuscript and looked at whether words ending in those suffixes clustered in the t-SNE plot of the Voynich word embeddings. We took clustering as evidence that that suffix was a morpheme, and lack of clustering as evidence that it was not.

The three groups of similar suffixes that we decided to focus on were:

1. *-or*, *-ar*
2. *-aiin*, *-ain*, *-oiin*, *-iiin*
3. *-edy*, *-ody*

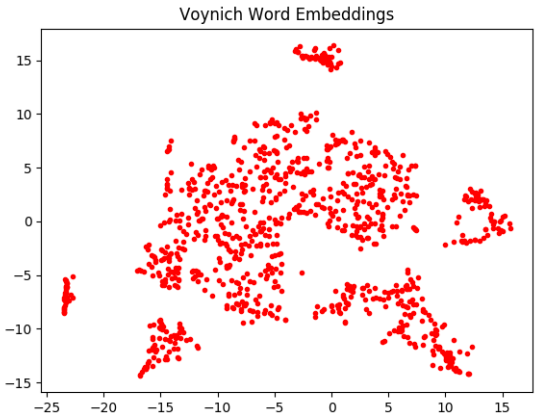

Figure 1: Voynich word embeddings. Each dot represents one word (with only the 980 most common words plotted).

Results of our analysis are discussed in Section 4.1.

3.2 t-SNE

[eb] The fastText vectors that we built from the Voynich manuscript have 100 dimensions. Such vectors are easy for computers to work with but difficult for humans to visualize. Therefore, it makes sense to apply dimensionality reduction algorithms to the data for visualization purposes.

The full name of the algorithm we used, which has become a standard in the machine learning community, is *t*-distribution Stochastic Neighbor Embedding (van der Maaten and Hinton, 2008). t-SNE works by computing the distance between each high-dimensional vector and tries to replicate those distances in low-dimensional space. Thus, nearby 100-dimensional vectors will be represented as nearby points; clusters of 100-dimensional vectors will become clusters of points.

It is important to note, however, that some information is lost in this reduction; there is no way to perfectly represent 100 dimensions in 2, in the same way that looking at a printed photograph edge-on (one dimension) will never provide a faithful reproduction of the original three-dimensional scene. While t-SNE is generally quite good at maintaining clusters of word vectors, it is a stochastic process that can introduce noise into the output. Thus, a cluster in t-SNE output is a strong indication, but not proof, of a corresponding clus-

ter in the original vector space. Applying clustering algorithms to the unreduced high dimensional data is an alternate approach that can yield more certainty at the expense of interpretability.

Figure 1 shows an example of the t-SNE output for the VMS. The clustering of certain points will be discussed below.

3.3 Topic Modeling

[eb] A natural first step for trying to decipher the VMS has been to find (near-)unique words on certain pages. If we are willing to accept that the text (or perhaps just labels) on each herbal page, for example, probably relates to the picture of that plant, then uncommon or unique words that appear nearby might describe the plant in question. Granted, this is an unproven assumption, so we must be slightly skeptical of any results produced using this method.

To formalize this process, we used a statistic know as tf-idf, short for *term frequency-inverse document frequency*. The *term frequency* counts how many times a certain term appears on one page; the *inverse document frequency* measures the occurrence of this term over all pages. tf-idf is simply the product of these two components:

$$\text{tfidf}(t, p) = \text{tf}(t, p) \cdot \text{idf}(t) \tag{2}$$

for some term t and page p.

A term that appears many times on one page, but on few other pages, will have a high tf-idf, while a term that appears across many pages will have a low score. For example, proper nouns would probably score higher than function words.

Various normalization schemes exist to remove biases towards long documents. We computed the term frequency as the count of the term on a page divided by the number of words on the page. The inverse document frequency is

$$\text{idf}(t) = \log \frac{N}{|\{p : t \in p\}|} \tag{3}$$

where t is the term, N is the number of pages, and the denominator counts how many pages contain term t.

One barrier we immediately ran into using tf-idf is the odd word distribution of the manuscript. Of the approximately 8100 words in the vocabulary, 5600 appear only once. Thus, most pages have unique words which are given high tf-idf scores. We can attempt to filter our results by removing single-occurrence words, but this risks throwing away useful data: perhaps the word for a certain plant or star really does only appear once.

While we cannot extract semantic content from tf-idf scores, the metric does provide a general direction for investigating words. It is useful to think about the results returned through this method as denoting words with high specificity or statistical importance on a page. Whether this correlates with semantic content remains to be seen.

The highest tf-idf words in the entire manuscript are on f65r, a page with one plant illustration and three words (*otaim dam alam*), all of which are relatively uncommon in the rest of the manuscript. We can therefore expect at least one of these three words to be highly specific to the illustration (again, assuming that the illustrations indeed match the text). Other high-scoring words include *cthaiin* (which occurs twice on f25r but only once on each of another 12 folios) and *chaiin* (which occurs once or twice on 33 folios, but four times on f4r). But alone, these high-scoring words do not appear particularly important; with so many unique words, we cannot expect to see many incredibly specific but commonly-occurring words.

3.4 Folio Vectors

[eb] The tf-idf metric also opens a new avenue into the creation of *folio vectors*. Just as we can create vector representations of words, we can do the same with each folio of the manuscript. Then we can look at how the folios cluster together.

As has been mentioned, Bunn (2017) attempted a similar analysis using a bag of characters representation for folio vectors. The issue with creating folio vectors solely out of character counts is that this sacrifices the potentially rich information encoded in word embeddings. Therefore, we instead decided to create folio vectors using our word vectors. We build a folio vector by taking a weighted average over all the words on the folio, where each word is weighted by its tf-idf score on that folio. In principle, this method should be able to represent a more nuanced notion of folio topic than Bunn (2017)'s bag of words approach. Once these folio vectors have been constructed, we can run similar analyses as with word vectors: we can look for clusters of similar pages with t-SNE or determine the most similar pages by cosine distance. Results of our folio vector analysis are discussed in Section 4.2.

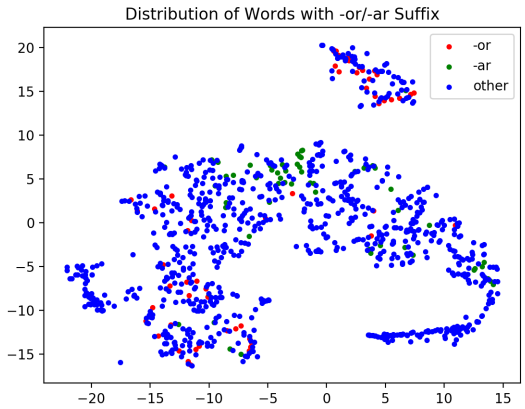

Figure 2: The scattered distribution of *-or* and *-ar* suffixes suggests that these suffixes are not morphemes.

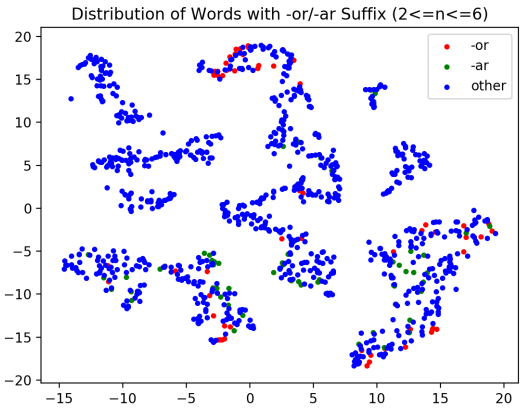

Figure 3: Including 2-grams in the fastText model produced a scattered distribution similar to that of Figure 2.

One issue with this approach, however, is that we can only build folio vectors from existing word vectors; uncommon words that have not been embedded into the vector space must necessarily be ignored. In our case, this means all words appearing less than five times overall are not counted.

3.5 MUSE

[wm] After building word embeddings from Voynich, we also built word embeddings from medieval and classical manuscripts that we obtained from the Perseus Digital Library (Crane, 2018). Our selection of texts spanned several genres, including alchemical and magical works, religious texts, and political treatises. The texts were written in English, Latin, Greek, Spanish, and Arabic.

We then ran the MUSE word alignment algorithm between the Voynich manuscript and each text. Once the model trained, we computed the closest alignments for each Voynich word ranked by cosine similarity and also visualized the aligned vector spaces in two dimensions using t-SNE. Results for these alignments are discussed in Section 4.3.

4 Results

4.1 Morphosyntactic Clustering

[wm] As can be seen in Figure 2, the suffixes *-or* and *-ar* were scattered among many other words in the t-SNE-reduced vector space. We took this as evidence that these suffixes do not carry morphosyntactic information. To verify that these results were not coming from the fact that the minimum n-gram length used by the fastText model was 3 (whereas the length of these two suffixes was 2), we reran the same analysis with a fastText model that included 2-grams. As Figure 3 shows, the resulting distribution looks, if anything, more scattered.

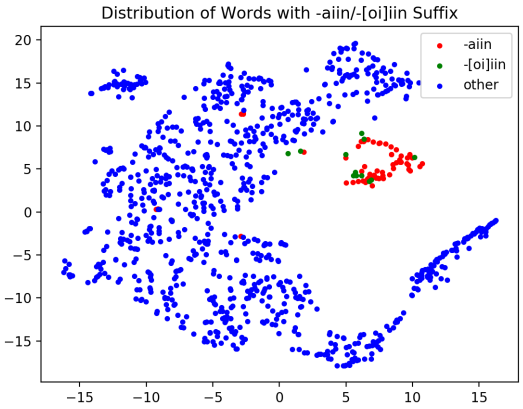

Figure 4: The strong clustering of *-aiin*, *-oiin*, and *-iiin* suggests that these suffixes reflect the same morpheme.

On the other hand, we found evidence that *-iin* is a morpheme in Figure 4. There is a distinguished cluster of words ending in the frequently occurring suffix *-aiin*. Additionally, most of the other occurrences of an *-iin* suffix, which are represented by the cases *-oiin* and *-iiin*, also appear in this cluster.

Figure 5 suggests that the commonly occuring suffix *-ain* may also be a morpheme, but, since its cluster is disjoint from *-aiin*, it does not carry the same information about a word's morphosyntactic

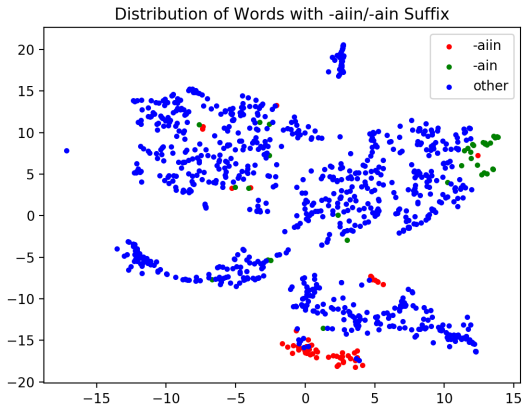

Figure 5: The suffixes *-aiin* and *-ain* appear to be different morphemes.

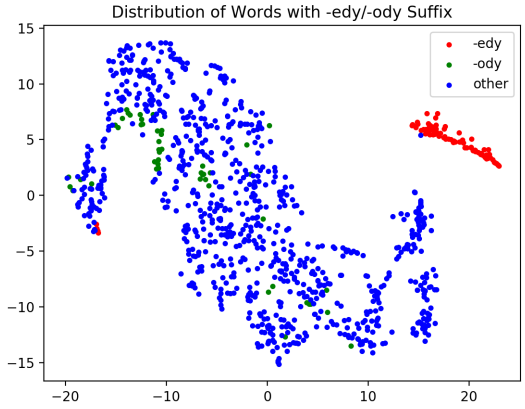

Figure 6: The suffixes *-edy* and *-ody* seem to appear in different contexts.

distribution as *-aiin*/*-iin*. In other words, it seems that these two common suffixes may be two different morphemes.

Finally, Figure 6 shows strong clustering for *-edy*, which we take as evidence that *-edy* is a morpheme. Words ending in *-ody* do not appear in the *-edy* cluster and display loose, if any clustering, among themselves.

These results are interesting in the context of Currier (1976)'s theory that *-edy*, which occurs overwhelmingly in Language B folios, and *-ody*, which occurs mostly in Language A folios, represent variants of the same morpheme in two different languages or dialects. Under this interpretation, *-edy* and *-ody* would have different distribution because *-edy* could only occur with other words from Language B and vice versa for *-ody*. We do find that *-edy* and *-ody* have different distribution, but our data does not provide evidence that *-ody* is a morpheme like Currier's theory claims.

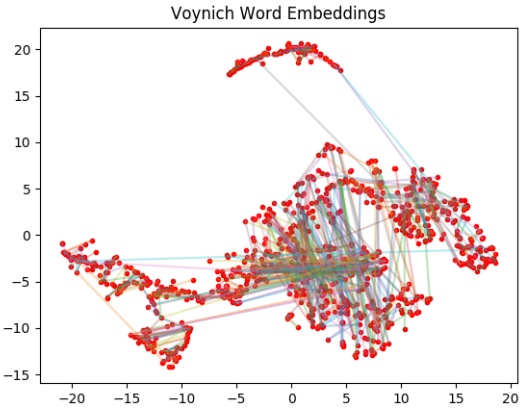

Figure 7: Voynich word vector pairings. From each point is a line to its nearest neighbor. Note that while some notion of clustering remain, other points have neighbors on the other side of the graph. This is the nature of dimensionality reduction.

[eb] We can also calculate the closest vector to every other vector. Figure 7 shows these connections visually, and some results are produced in Table 1. A variety of presumptive affixes are very apparent: *qol* vs *ol* vs *sol*, for example. We can also see that certain letters are closely related and perhaps interchangeable with others. It is important to remember that these vectors were computed based on the word's context; the fact that *qolchedy* aligns with *olchedy* suggests that they appear in similar syntactic (or even semantic) positions and not just that they have similar spellings.

4.2 Topic Modeling

[eb] Our analysis of folio vectors gives some interesting insights into topic modeling in the manuscript. We observed some clustering among vectors in the bath section, and a slight bias in the astronomical and multiherbal (herbal pages containing many small plants, rather than one large plant).

Using word vectors and tf-idf to construct folio vectors is particularly powerful as an analytic tool. Word vectors are very local in scope: they take into account a word's individual letter sequence, as well as the context of words it appears within. tf-idf, on the other hand, treats each page as a bag of words, ignoring the ordering and distribution of words, and instead tries to assess how important a

qolchedy	olchedy
polchedy	solchedy
oraiin	soraiin
olkeedy	solkeedy
opchedy	ypchedy
chckhhy	chckhy
tchody	ytchody
tshedy	kshedy
todaiin	podaiin
ykchedy	olkchedy
oraiin	araiin
tcheody	kcheody
pchodaiin	shodaiin
ofchedy	ypchedy
ochedy	ychedy
ksheody	psheody
dshedy	tshedy
okalal	okalar
qolchedy	solchedy
qolchedy	polchedy
araiin	raraiin
osaiin	saiin
olcheol	lcheol
podaiin	odaiin
opchdy	ypchdy
yshedy	lshedy

Table 1: Top closest word vector pairs (in descending order of distance) Morphological similarity is incredibly apparent.

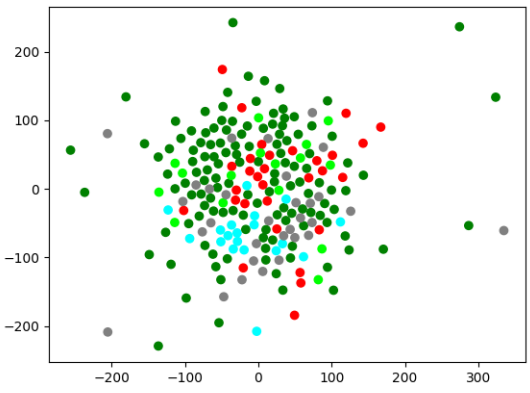

Figure 8: Document vectors colored by topic. Cyan: bath; grey: text; dark green: herbal; light green: multiherbal; red: astrological.

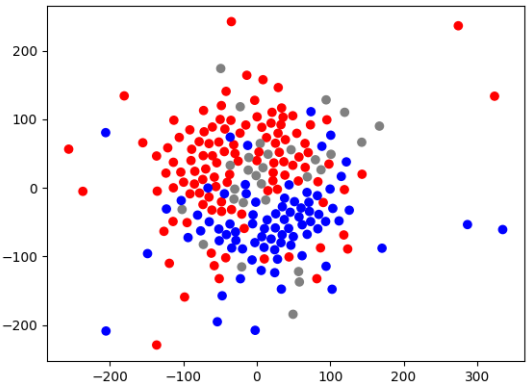

Figure 9: Folio vectors labeled by Currier language. Language A is red, and language B is blue.

word is *globally*.

While these are not strong results, the fact that we see clustering at all among the folio vectors suggests that we have found words that are both related to each other, and highly specific within their respective folia. That is, we see *similar important words* in each section (where "importance" is as measured by tf-idf). While we would remiss to call this a topic in the semantic sense, we nonetheless observe a kind of morphosyntactic "topic" (for example, in the bath section, *qo-* words have very high tf-idf scores). Whether this is due to true semantic differences or simply different word patterns is beyond the scope of this model.

We also used folio vectors to investigate Currier's language theory. As was the case in (Bunn, 2017), we observed some segmentation of the data into different languages, but no definitive clustering – the folio vectors do not provide evidence for Currier's labeling (Figure 9). In general, our folio vectors cluster into a single large group, so we cannot visually identify any meaningful groups from the t-SNE plot.

4.3 Unsupervised Word Alignment

[eb] Unfortunately, MUSE did not give us the same dazzling results with Voynich as it did with the large corpora used in the original paper. The algorithm was largely unable to align our Voynich vectors with those of many other works in a number of languages. Oddly, we often found that MUSE aligned many Voynich words to the same word in another language, and on t-SNE plots of the resulting alignments, we generally found

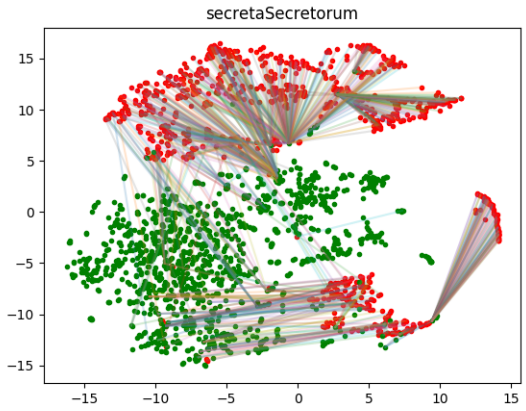

Figure 10: Mapping visualization between VMS (red) and Secreta Seretorum (green). Each lines connects one Voynich word with its alleged Latin counterpart.

two mutually exclusive clusters: one of Voynich words, and one of the other text's words (instead of the overlapping clusters we expected).

Since we did not write the code, and instead only repurposed it to work with our embeddings, it is hard to say exactly why MUSE did not work. However, the original paper (Lample et al., 2017) used high-quality word vectors trained on the entirety of Wikipedia (nearly 2 *billion* words), and then further selected the 200,000 most common words for testing. This is many orders of magnitude more training and test data that we had; it is perhaps unsurprising that this experiment did not perform well.

When we looked closely at many of the alignments, it seemed like similar Voynich words were being mapped to similar (or the same) target language word. In a sense, this is good; this is the general direction we would need to go for MUSE to work. However, this is not a new finding. What we are seeing, most likely, is simply clusters of Voynich vectors mapping to the same nearby Latin word. This is something we can easily visualized in t-SNE: in Figure 10, each line maps a Voynich word to its best Latin match. While our usual notions of distance are skewed by the dimensionality reduction (each line is drawn to the "nearest" point), the common alignment of large clusters is clearly visible.

MUSE was therefore unhelpful in providing unsupervised word alignments in Voynichese; we are severely limited by the size of our data set and the quality of our word vectors. However, all hope is not lost. This general approach might have some prospects for the manuscript, but would need additional linguistic and computational work.

5 Conclusion

[eb] Once again, the mysterious Voynich Manuscript has evaded decryption. Although we had no luck creating unsupervised Voynich word alignments, we have unearthed a number of fascinating morphosyntactic phenomena by creating word vectors for the manuscript. We identified possible affixes by looking at vector clustering, and there are, of course, many more to investigate. This is an incredibly exciting result; further work in this direction may lead to some semblance of morphological understanding of Voynichese.

While our morphological and topical analyses picked up on structural properties of suffixes and important words in the text, this does not necessarily mean that the text is written in a natural language. Formal language theory tells us that many sequences of structured text can be described by a grammar. Therefore, it is always possible that the properties our embeddings associate with specific suffixes or folia reflect non-natural-language structures or gibberish.

6 Further Work

[eb] Word vectors present many opportunities for experimenting with the relationships between words. A common demonstration of word vectors is to highlight their ability to detect analogies. Vector equations such as *queen* – *woman* + *man* ≈ *king* show that word vectors can encode a surprising amount of information (in such equations, we add and subtract the vectors, and then search for a nearest match to the result). Therefore, an avenue for further research would be to try and replicate this phenomenon in the VMS. Specifically, it might make sense to test morphological analogies like *tchedy* – *tchody* ≈ *kchedy* – *kchody*. A more ambitious goal would be to try to use the same notion of analogy to tease at semantic relationships between words.

The Takahashi transcript, while consistent and complete, is nearly twenty years old. Now that extremely high-definition scans of every page of the manuscript are available from the Beinecke, it is high time for an updated EVA transcript. While

there definitely are some unreadable words, the new scans make easy what was impossible on old photocopies of the manuscript.

And more work is in store for the Voynich alphabet. While we have been relying on EVA as the golden standard, this is not necessarily a valid assumption. A neural network could be trained on individual, maximally decomposed stroke data, and then try to classify groups of strokes into characters. This might be doable with clustering, or by taking advantage of Zipf's law to model unigram or bigram letter frequencies. Perhaps such efforts to improve transcriptions of the manuscript would also help in the creation of higher-quality word vectors.

References

Julian Bunn. 2017. Using t-distributed stochastic neighbor embedding (tsne) to cluster folios. https://voynichattacks.wordpress.com/2017/09/26/using-t-distributed-stochastic-neighbor-embedding-tsne-to-cluster-folios/.

Ed. Gregory R. Crane. 2018. Perseus digital library. (accessed May 6 2018). http://www.perseus.tufts.edu.

Prescott H Currier. 1976. Some important new statistical findings. In *Proceedings of a Seminar held on 30th November*.

John R. Firth. 1957. A synopsis of linguistic theory. *Studies in linguistic analysis* .

Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043* .

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR* abs/1301.3781. http://arxiv.org/abs/1301.3781.

Christian Perone. 2016. Voynich manuscript: Word vectors and t-sne visualization of some patterns. http://blog.christianperone.com/2016/01/voynich-manuscript-word-vectors-and-t-sne-visualization-of-some-patterns/.

L.J.P. van der Maaten and G.E. Hinton. 2008. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research* .

Bayu Aldi Yansyah. 2017-. fasttext: A python interface for facebook fasttext library. (accessed May 6, 2018). https://pypi.org/project/fasttext/.