Final Project Write-up

Applying Network Science Methods to Semantic Text Analysis for Categorization of Sentiment in Amazon Product Reviews

CSC-395 Network Science

Written by

Caio Carnauba and Mira Tellegen

October 21, 2020

Although computer science is often thought of as a field focused on numbers, writing programs that are capable of understanding human language has been a major focus in the field. In recent years, network science methods have arisen in the field of semantic text analysis as ways to improve the speed and accuracy of the analysis. Researchers find network science helpful to categorize and analyze text data when the data inputted is complex, unprocessed, or does not follow clear categorization rules. In our work, we focused on semantic text analysis using a network science approach. The algorithm that we explored took a data set of strings, then transformed it into a network where each node was one of the text fragments from the data set. In the network, two nodes were adjacent if they were considered similar based on criteria meant to evaluate the sentiment of the nodes. We expected that the communities in the resulting network would represent different sentiments. By analyzing the network, we hoped to gain additional insight on the data set which would not be possible when simply reading the text. Furthermore, since text analysis isn't commonly connected with network science, we were interested in the application of network methods to natural language text.

We wanted to explore the idea of sentiment analysis by creating a semantic network of a data set of Amazon product review titles. We hoped to reproduce the results seen in the video "Practical Graph Theory: Applications to Real World Problems with Python", by applying the method to a product review data set we found, instead of the employment data set used in the video. [5] Our literature review revealed that semantic analysis often focuses on data sets of large text, so we were interested in analyzing short user inputted text like the product reviews. The sentiment analysis performed in the video "Practical Graph Theory: Applications to Real World Problems with Python" seemed well suited to product reviews, since the reviews inherently expressed a sentiment, and since the product reviews mirrored the length and style of the data set from the original video. [5] The video inspired our primary research question of whether we could use network science semantic text analysis techniques to accurately categorize the sentiment of Amazon product review titles.

Exploring text analysis through network science and Julia was an interesting approach because Julia is a language with a lot of math and network functionality, but fewer methods focused on string analysis. We were very interested in performing string analysis in Julia because it would take advantage of Julia's ability to process large data sets as an expansion and new application of the Python method from the video. [5] We were also intrigued to work with short strings that were written by users, where the text contains fewer characters to analyze. With texts that have very few characters expressing their sentiment, the similarity comparison of the texts may

not vary as much as with longer texts, which could affect the complexity of the semantic network. We believed that using a network to find neighbors and communities between texts could prove very effective in identifying similar sentiments, regardless of the length of the input, since network methods allowed us to examine similarity links without relying solely on direct connections between texts.

Before diving into the project, we researched previous work in the field, focusing on semantic text analysis and network science text analysis. Our literature review allowed us to plan our project with a full understanding of previous research methods that combined network science methods with text analysis goals. We found that the network science methods in the research varied widely, but most papers used some common building blocks for their experiments.

To contextualize these common threads between research approaches, we examined a paper by Phillip Drieger that laid out the main definitions and terminology used in network science text analysis. Primarily, Drieger extensively defined semantic text analysis and semantic networks. A semantic network is a network where nodes represent text fragments in a data set and edges represent the similarity between those texts. Some semantic networks are two-mode, where one set of nodes correspond to text fragments, and the other set of nodes correspond to the texts themselves. Semantic analysis is a subgroup of automated network analysis where network statistics are used to categorize natural language text data based on criteria set by the researcher. [4]

It's important at this point to also define another key term in semantic text analysis, a knowledge base or ontology. Most text analysis methods rely on a knowledge base, sometimes referred to as an ontology, which is often a thesaurus or structure that records categories to associate with different texts in the data set. Text fragments, or nodes, can be compared to the ontological categories and compared to the texts that have already been categorized. The nodes are linked by "semantic relations" based on their relations to the ontology. [11]

All of the research papers we examined used semantic networks to find results about their data set and many of them used ontologies to build their semantic networks. When it came to analyzing the semantic network, the papers began to deviate in the method. Overall, the research goals of the papers fell into two categories: those which proposed novel methods to semantic text analysis, and those which used existing methods to achieve new results.

Papers expanding existing text analysis methods or inventing new methods often

shed light on existing issues in the field of network science text analysis, which we found very helpful in assessing the pros and cons of our method choices. Two such research papers we found focused on training and analyzing new neural network models to rank similarities of texts, as a more versatile method than existing work. In a paper by Kiran Mysore Ravi et al., they trained a Long Short Term Memory variation on an RNN model to analyze unprocessed raw text, which allowed them to analyze diverse text datasets with a central method. [8] Similarly, in a paper by Chanzheng Fu et al., the researchers evaluated their new memory neural network model, which outperformed an existing neural network variation. [6] However, whereas Ravi et al. used n-grams to rank similarity in the text, Fu et al. deviate from the n-grams method, which they believe is becoming less relevant as network science methods improve. [8] [6] Our research is more similar to the work of Ravi since we also worked with raw text and examining it through k-grams. We became interested in their work with neural networks as a more effective similarity ranking, since we struggled with our similarity algorithm throughout the project. However, in an effort to limit the scope of our project, we did not incorporate any neural network methods into our method.

The novel analysis methods proposed in a paper by Livia Celardo et al. focused on experimenting with cluster analysis of the semantic network. We adjusted our network analysis process significantly throughout the project, so Celardo et al.'s work on improving analysis accuracy related to our struggles with creating realistic keyword clusters from our network. Celardo et al. aimed to improve analysis accuracy by modeling data more realistically with the incorporation of text co-clusters. Whereas current models often create network clusters where the mean value converges toward the cluster center, these researchers expanded the text clustering methods by partitioning both the rows and columns in the matrix of similarities. Since our project relies significantly on the manipulation of kernel matrices containing our text similarities, we found that their work with matrices provided helpful context for our matrix manipulation. [2]

The most surprising new research we examined was in a paper by Mattea Chinazzi et al., where they deviated from the norm of using an ontology, instead comparing the similarity of texts using an n-dimensional vector space. All other papers we examined relied on knowledge bases to rank text similarities, as does our method, so their research stood out from the body of work we examined. Chinazzi et al. ranked text similarity based on the texts' closeness in the vector space, and were then able to create a Research Space Network that mapped taxonomies of the dataset. So, they were able to effectively categorize text without starting with an ontology of the data taxonomy categories. [3]

The examination of new ideas in semantic text analysis allowed us to develop a deeper understanding of our work and creatively problem solve in our process, but we decided to focus on another type of research in connection to our project goals. Namely, a significant portion of the sources in our review took new data sets or subject areas and applied existing network science techniques to the semantic networks for more complex text categorization.

For example, many research papers we read relied on relating data sets to thesauri ontologies to determine similarities and edges in the network. In a paper by Roberto Willrich et al., they performed this type of knowledge base analysis to determine students' reading comprehension of the text, which is a type of sentiment analysis. [11] Similarly, in a paper by Manuel W Bickel, the researchers used text mining on large climate action plans, and related the resulting data set to three knowledge bases to analyze climate action plans by known methods. The researchers also used multiple types of similarity matrices, called "document section term matrices" and "document category term matrices", to consider gaps in current climate action. [1]

Researchers also often applied common network analysis techniques to their text datasets and semantic networks to discover complex categorizations of the texts. In a paper by Sang M. Lee and Rha Jin Sung, they analyzed their semantic network of research in the service industry by using centrality and clustering statistics, to discern topic categorizations in their data set of service industry research papers. [7] Similarly, in a paper by Filipi N. Silva et al., they started with a keyword framework knowledge base of taxonomies, but used shortest path lengths to find similarities in the resultant semantic network, and used their results to identify taxonomies in the data set through semantic connections. [9]

Although many researchers used similar methods to ours, one paper stood out as relating particularly closely to our project goals. In a paper by Herman Wandabwa et al., the researchers focused on applying a deep convolution neural network framework to the text analysis of an uncommon text type. They used short user inputted text streams in the form of Tweets, as opposed to the longer texts that are common in semantic text analysis. We decided to also focus on particularly short texts, in the form of Amazon Product Review Titles, which are user inputted text under 20-30 characters. This type of text poses unique challenges when translating to a semantic network, because there are very few characters, so we have fewer k-grams to compare, which refers to sub-strings of length k. [10] We want to examine and apply the semantic text analysis methods from a video by Tyler Foxworthy, "Practical Graph Theory: Applications to Real World Problems with Python", who presented

a method designed for short user-inputted text. [5] We decided to examine our data set of short titles of Amazon product reviews by creating a semantic network with Foxworthy's steps, with the ultimate goal of categorizing them based on sentiment.

The method from Foxworthy's video had six main steps. First, Foxworthy preprocessed his dataset to remove white-space and punctuation. Then, he used k-grams to create a feature space of all possible k-grams in the alphabet. This feature space acted as the knowledge base in Foxworthy's method. He then "vectorized" each text in the data set by creating vectors of zeros the size of the feature space that correspond to each text, and marking a 1 at each vector index where the string contained the k-gram corresponding to that index. To track the similarity of the vectorized texts, Foxworthy used a similarity algorithm called hamming distance, where two vectorized strings are compared to each other, and the distance between them is the number of indices at which the vectors are different. The hamming distances were stored in a kernel matrix, where each row or column represented a text in the data set, and their corresponding index was the similarity between the texts. Foxworthy found a "cutoff" value through taking the eigenvector of the kernel matrix, and created his network by marking an edge in an adjacency matrix for each pair of texts whose hamming similarity value was above the cutoff. The adjacency matrix corresponded to a semantic network from which Foxworthy extracted communities and sentiment keywords to characterize the communities. [5]

We started by following the steps of Foxworthy's method, but customized it more and more to our data set as the project went on. Our testing of Foxworthy's methods and experimenting led us to adjust our steps in response to errors in the process, or from practical concerns about using a different data set and coding language than Foxworthy.

Our first step in our project was to preprocess our data set of Amazon product reviews. The original data set included the full reviews, but Foxworthy's method in the video was for very short text fragments, so we elected to attempt to categorize the sentiment of the reviews by analyzing a data set of their titles. [5] We read in the data removed whitespace and punctuation, then downcased all the entries. One of our first major stumbles in the project came with the decision whether to examine n-grams of words in the titles or k-grams of characters in the titles. As we discussed above, much of the existing research concerning network science text analysis used n-grams. However, most research we found examined long texts with lots of words, whereas Foxworthy's video analyzing short texts used k-grams. We initially wrote and tested functions to split the data set entries into both n-grams and k-grams. However, as we examined how short the reviews were, many entries were only one or

two words, so it made more sense to proceed with the k-grams functions. We also created a knowledge base/ontology for our work, in the form of a feature space where each possible k-gram in the alphabet was recorded. Initially, we implemented a way to create both a k-grams feature space of all possible k-grams in the alphabet, and an n-grams feature-space that recorded each possible n-gram of words in our data set. Writing the code to create an ontology of all possible n-grams of words in the data set assured us that k-grams made more sense for the project, since a k-gram ontology is limited by 27 characters, the space character and the alphabet, whereas there could be a very large ontology created from considering all the different words in a given data set.

After deciding on k-grams, the next functions we implemented were similarity functions to assess similarity of different data set entries. Initially, we didn't consider that our similarity function would need to examine vectorized strings instead of the string literals from the data set. Our first implementation to calculate similarity was a type of edit distance function which compared two strings based on character-to-character difference. After testing, this similarity function worked to precisely calculate the similarity of strings through one-grams/characters, but was not useful in our ultimate goal of comparing vectorized strings by k-grams. In our adjusted function, we implemented a hamming distance algorithm, where the hamming value would reflect the number of indices in which the vectorized strings differed. Speaking in terms of k-grams, we outputted the number of k-grams that differed between the strings. The hamming algorithm was a challenging implementation, since at this point we had not written code to vectorize our data set, which meant the function was written before we had test cases.

To vectorize the data set, we combined our earlier functions to preprocess our data set, to compare each string to the feature space, and to create a vector based on the k-grams it contained. This allowed us to test our hamming distance function, which matched Foxworthy's work. However, at this point we had concerns about runtime, since our data set was very large and we were beginning to work on large matrix and network manipulations in the method.

This concern proved valid when we implemented a function to create a kernel matrix. In that function, we calculated the hamming distance for each pair of texts in the data set. Since the data set was initially 14326 elements, and the k-gram feature space for a bi-gram is 729 elements and the feature space for a tri-gram is 19683 elements, the function compared 205,219,950 pairs of vectors where every vector had either 729 or 19683 indices to examine. Although we were able to build a kernel matrix in the bigrams case with a runtime of 154.318348 seconds, in the trigrams

case, the runtime was so long that we terminated the attempt to calculate it after an hour, and we began to have serious concerns about other functions operating on such a large data set and featurespace. At this point, we decided to cut down the data set to limit string size. We tested strings sized between 15-30, and decided that bi-grams and tri-grams both had a reasonable runtime when we capped the data set at 25 or 30 character length. Using a 25 character cap, the data set was 10938 elements and the kernel creation using bigrams took 82.976001 seconds, whereas trigrams took 1383.577786 seconds. We also ended up using the Julia built in hamming distance function, which had a runtime of 0.000825 seconds per pair of vectorized texts as opposed to our functions' 0.093454 seconds per pair. We hypothesized that that small efficiency boost could multiply as we assessed many string pairs, so made the switch in our implementation, although we saved our similarity algorithms in the code.

With the runtime issue partially resolved, we examined how to translate the kernel matrix into an adjacency matrix. Foxworthy used a cutoff value, where he put an edge between texts with a lower hamming similarity value than the cutoff. Since hamming distance counts the differences, two vectorized strings that are identical will have a hamming distance of 0. We attempted to implement Foxworthy's method, which involved finding the second eigenvector of the kernel matrix, then taking the sum of the eigenvector over the length, however, this consistently gave us negative cutoff values between 0 and -1. [5] Therefore, there were no texts that had a hamming value less than the cutoff. This posed a serious issue in creating the network, since we didn't want to pick an arbitrary cutoff, but we also couldn't use our version of Foxworthy's implementation. We eventually scatter-plotted the hamming distances from the kernel matrix, and selected cutoffs based on the distribution. Running some examples, we thought it was more intuitive to change our hamming distance function to track hamming similarity, and count the number of indices that vectors were similar. This way, we could choose cutoffs that were higher on the scatter-plot and further the intuitive sense that a high hamming value means high similarity.

Our cutoff method allowed us to translate our kernel matrix into an adjacency matrix, and translate that into a semantic network. In the analysis phase of the method, we started by examining the neighbors of nodes in the network, and we were able to tweak the cutoff by seeing that at very high cutoffs, all neighbors of a random node were identical words, but with slightly lower cutoffs, we attained our goal of the neighbors of a node being similar words. We considered examining just the neighbors close to given nodes to see similarities in the data set, but realized that that would be impractical and ineffective in finding the type of sentiment clusters we wanted, since the neighbors of a node don't necessarily express the nodes that are clustered with it. The video used network communities as the method to pull keywords, and

we realized that communities allowed us to examine reviews that were very related in the network without necessarily being close neighbors, which allowed for a more general view of semantic connections.

To pull communities from the network, we decided to use Julia's built-in label propagation function. Two flaws we encountered in the resultant communities were that the texts in the largest community didn't seem related, with titles like "good", "nice", and "sucks" or "lovely product" and "average" together in the same community. We also saw many communities that were similar to other communities in the network, such as a community with variants of "value for money" versus a community with variants of "value of money". We hypothesized that fluff words like "for" and "of" were separating communities that expressed the same sentiment, so we implemented a portion of preprocessing that removed fluff words like "for", "as", and "and". We hoped the function would merge some communities that were separate because of fluff word differences, and allow us to include longer data set entries without increasing runtime, since removing fluff words lowered the character counts.

Next, we ran the method on titles of 25 characters or less in the data set, using trigrams with a cutoff value of 19678, and found 460 communities containing more than one element. The table below includes some examples of keywords from some of the communities in the semantic network.

1	not worth it	not worth	not worthy
2	value money	value money	value money
3	not buy	do not buy	no not buy it
4	good headphones	nice headphones	good headphones
5	awesome product	awesome	awsome
6	wow	ok ok	nahhh

With these communities, we were able to discern reviewer sentiments such as advising other buyers, considering the value of money for the product, and rating its function. We were also able to visualize the network, which had some clear communities and some reviews that didn't meet our similarity criteria to be linked to other texts.

For most of the steps in our method, we fulfilled a goal without making decisions that introduce personal bias. For example, preprocessing the text simply made it easier to use in functions, it included no judgement or bias from us. Similarly, creating the kernel matrix just translated previous similarity data into a data structure,

without risk of bias. However, a few steps in the method introduced personal bias and judgement calls into the semantic network creation and analysis. We chose the cutoffs visually based on a scatter plot, so our personal judgement determined the number of edges in the network and how similar two texts need to be to have a semantic connection based on the appeal of the keywords in the clusters the cutoff created. Another area of personal bias was with the keyword selection. With many of the communities we saw, the reviews were very similar and keywords that appeared often were easily discernable. However, with clusters that had more variation, we selected keywords that seemed particularly indicative of the community, which could affect which results we were displaying.

Beyond the potential effects of biases, one large limitation of our work was that the method was designed for very short strings, and would have too large a run-time with larger texts. However, we would also consider this to be a strength, since strong network science methods already exist to analyze large texts, and our method focused on a less explored field of shorter texts. We could also imagine that our similarity function may have missed some very similar texts in cases of misspellings of the same words or phonetic matches. In the case of the misspelling "eydegess" and the word "edges", very few k-grams would match, despite the strings relating to the same word, so the hamming similarity would be small. Similarly, in the case of phonetic similarity between words, like the two spellings of the same name "ashlee" and "aishleigh", the hamming similarity would not reflect that the words are essentially the same when spoken. One way we could address this limitation would be to add another similarity test based on a phonetic dictionary, to check for review titles that are the same idea, but misspelled through user error.

We also discovered that the largest communities had many one or two word reviews which were not very related to each other, like the examples above of "wow" and "ok ok". We theorized that these types of one word judgements weren't long enough to be properly assessed in terms of trigrams, so were not necessarily linked to others with similar sentiments. A next step in refining our research would be to find ways to split the largest communities into smaller communities that reflected sentiment more effectively. We noticed that most of the texts in the largest community were one or two words, so one way to re-examine them would be to take a smaller data set of titles from the largest community, and analyze them as their own data set using bi-grams or one-grams. This might allow a more specific similarity comparison between the texts. Another solution would be to create a second knowledge base in the form of a thesaurus, with categories based on the type of one word judgements we see in the largest communities, like "good", "nice", and "bad". This would allow us to categorize one-word titles more precisely, based on sentiment categories. However,

creating this thesaurus would present another opportunity for our personal biases to affect the communities.

Other than the weakness of the largest community, we found that most of the communities clearly expressed a sentiment of the reviewers, whether that be monetary concerns, desire for good sound quality, or expressing that they would purchase the product again, or just general satisfaction or dissatisfaction. Another next step in refining these communities would be to develop a method for picking the most central review titles or keywords in the communities, to take the visual analysis aspect out of the keyword selection. Additionally, the communities were so effective that sometimes many of the reviews in the community were near identical. Incorporating different similarity requirements or experimenting with lower cutoffs could result in more diverse semantic communities. Therefore, we overall met our research goal of categorizing the data set by sentiment in a time-efficient way, but we could work towards a clearer and more objective categorization methods.

References

[1] M. W. Bickel, "A new approach to semantic sustainability assessment: text mining via network analysis revealing transition patterns in german municipal climate action plans," *Energy, Sustainability and Society*, vol. 7, pp. 1–25, 2017.

> This paper focused on text mining German climate actions plans to see patterns in the text networks. In the experiment, three thesauri described categories, then the researchers ranked these categories by their perceived network importance. This type of analysis is very similar to our experiments, since the researchers categorized sentiments in the climate action plans. So, we looked to this paper as a good example of sentiment analysis. An ontology also played a key role in this paper, when they translated a vector space model of "document-section-term-matrices" into "document-category-term-matrices" through relations to the ontological categories. Therefore, this paper showed the importance of matrices and models to determine links in a text analysis network. The researchers were able to highlight improvement areas in the climate action plans, including suggesting more renewable resources in the heat and mobility sectors.

[2] L. Celardo and M. G. Everett, "Network text analysis: A two-way classification approach," *Journal of Information Management*, vol. 51, pp. 1–8, 2020.

This paper proposed an expansion of the text clustering analysis method used in network semantic text analysis, using co-clustering. Clustering text can lead to clusters where the mean value converges toward the cluster center, which is rarely seen in real text data. Instead, the researchers simultaneously partitioned the rows and columns of matrices to create "co-clusters", and use a two-mode matrix in the place of the common space-vector model. As a result, their new method for community detection considered the texts and words simultaneously, both in the rows and columns of the affiliation matrices. They concluded that the co-clustering approach avoided the mean value convergence and therefore mirrored real data more closely. We included this research because of its innovative use of the matrix for text analysis, and because they focused on mirroring patterns in real text data. Since we worked with user-inputted review titles, our dataset may show patterns unique to natural language text.

[3] M. Chinazzi and et al., "Mapping the physics research space: a machine learning approach," *KSM Consulting*, vol. 8, no. 33, 2019.

The goals of this paper were very similar to the other paper we examined about scientific taxonomies. The researchers mapped scientific knowledge categories to be able to classify topics and taxonomies from the data. This paper suggested that the traditional text analysis methods that rely on knowledge bases of taxonomies can be restrictive. So, this research created a new categorization method, where they used n-dimensional vectors to represent scientific topics, then ranked their similarity based on how close they were in the n-dimensional space. By not relying on a taxonomy knowledge base, the researchers found that they could analyze a wide variety of scientific field with their model. We included this paper because their network analysis was very similar to the other text analysis papers we read, but focused more on the model, and less on the idea of semantic text analysis. We were interested in their expansion of analysis methods to be more versatile to different data sets.

[4] P. Drieger, "Semantic network analysis as a method for visual text analytics," *Procedia - Social and Behavioral Sciences*, vol. 79, pp. 4–17, 2013.

This paper broke down the definition of a semantic network and the idea behind semantic network analysis. The researchers spent time distinguishing semantic text analysis from automated network analysis,

where algorithms are used to compute statistics related to the network. Semantic network analysis is a subgroup of automated network analysis because network analysis techniques are used to categorize a semantic network of text fragments. The researchers also explained that sparse networks can indicate generally unrelated text fragments in the semantic networks, whereas dense networks represent coherent texts with lots of links between words. Their experiments used the degree distribution and clustering statistics to categorize the text in the semantic network, and found that networks can improve efficiency in text analysis. We appreciated the definition and breakdown of the basics of the field of network text analysis, and we relied on this paper as the basis of our description of semantic text analysis.

[5] T. Foxworthy, "Practical graph theory: Applications to real world problems with python," *KSM Consulting*, 2015.

In this video, the presenter explained how to use network science to categorize the sentiment of unclean text, specifically a data-set of user responses to an employment survey. Their data-set was pre-processed, then vectorized with an n-grams method. Then, a kernel matrix was used to categorize the similarities between different text fragments. This allowed the researchers to create a network and find communities of similar texts, then categorize the sentiment of the text fragments in the network. As a result of this method, the researchers were able to use network science to process short text fragments and to automate the processing of many data submissions. We included this source because we replicated this method to categorize the sentiment of the Amazon product reviews in our data-set. We were very interested in the application of network science to very short text fragment entries to a data-set, where there are only a few keywords in each node driving the analysis.

[6] C. Fu and et al., "Sememnn: A semantic matrix-based memory neural network for text classification," *The Institute of Electrical and Electronics Engineers, Inc. (IEEE) Conference Proceedings*, pp. 123–127, 2020.

These researchers adapted the existing Memory Neural Network model (MemNN) to create a Semantic Memory Neural Network (SeMemNN) for use in semantic text analysis. They evaluated their new model on different configurations, exploring the breadth of text analysis. The

researchers applied different Long Short Term Memory model configurations to their SeMemNN, including configurations double-layer LSTM, one-layer bi-directional LSTM, one-layer bi-directional LSTM with self-attention. They found that their novel model outperformed VDCNN, an existing neural network option. We chose this article for its description of how methods of text analysis evolve. For example, this article suggested that text analysis is moving away from a bag of n-gram linear vector methods, since network science models allow for accurate analysis without n-grams.

[7] S. M. Lee and R. J. Sung, "A network text analysis of published papers in service business, 2007–2017: research trends in the service sector," *Service Business*, vol. 12, no. 4, pp. 809–831, 2018.

The network text analysis performed in the paper focused on the analysis of clusters in the network to identify central topics in the service industry. The researchers applied clustering and centrality statistics to a network created by text mining and examine the structural-semantic relationships in the network. This paper also displayed an application of matrices, to store the co-occurrence frequency of texts. They suggested PageRank as a future method to include the importance of different texts in the network. As a result, they were able to quantify the balance between traditional topics and innovative topics in service industry research, which could be useful to future researchers. [5]

[8] K. M. Ravi, J. Mori, and I. Sakata, "Cross-domain academic paper recommendation by semantic linkage approach using text analysis and recurrent neural networks," *The Institute of Electrical and Electronics Engineers, Inc. (IEEE) Conference Proceedings*, pp. 1–10, 2017.

This research focused on the use of a Long Short Term Memory variation on the RNN model. The researchers used their text analysis model to make recommendations of similar texts, which they then ranked based on semantic similarity. With their trained model, the researchers found that unprocessed, untagged raw text could be analyzed without any algorithm to help interpret the natural language. We found that Long Short Term Memory models were used in numerous other recent text analysis experiments. However, whereas other papers indicated that the bag of n-grams text analysis was not ideal, these researchers used a bag of n-grams in tandem with network science. We were interested in this research because our guiding video, "Practical

Graph Theory: Applications to Real World Problems with Python", uses k-grams as a preliminary step in the network analysis, and we explored n-grams as an initial option. [5]

[9] F. N. Silva and et al., "Using network science and text analytics to produce surveys in a scientific topic," *Journal of Informetrics*, 2016.

These researchers applied an importance index to a citation network generated through the Web of Science to create a keyword framework of taxonomy in scientific fields. The shortest path lengths of the network were the determining factor in the network analysis, since the researchers used shortest path lengths between keywords to find strongly connected components within the network. Therefore, the shortest path statistics determined the clustering and eventual categorization of the text. The researchers found that their network accurately expressed scientific taxonomies, and that border communities in the network revealed interested subcategories of the data. We were interested in the shortest path length application here as a way to categorize the relationship between nodes. Furthermore, the result of keywords drawn from the network communities paralleled our goal of finding sentiment keywords in the reviews.

[10] H. Wandabwa, M. A. Naeem, and F. Mirza, "Document level semantic comprehension of noisy text streams via convolutional neural networks," *The Institute of Electrical and Electronics Engineers, Inc*, pp. 475–479, 2017.

These researchers conceptualized a network framework to perform analysis on native language text in short data streams and text messages like tweets. Many of the current network science interpretation models can't process short data streams like tweets, where incomplete words and slang are common, so these researchers expanded the model. The researchers designed a deep convolution neural network framework, and found that the network was able to analyze slang words and Twitter-specific linguistic patterns on very short text inputs. Since much of the research in text analysis is analyzing large documents in a time-efficient way, we chose this research for its analysis of short text streams. Our review titles are text fragments, so this paper's data-set most closely aligns with our intended data.

[11] R. Willrich and et al., "Capture and visualization of text understanding through semantic annotations and semantic networks for teaching and learning," *Journal of Information Science*, vol. 46, no. 4, pp. 528–543, 2020.

In this paper, the researchers assessed the reading comprehension of texts in classrooms by matching students' annotated texts to a knowledge base. By tracking text annotations in semantic networks, the researchers found that teachers could assess student comprehension more quickly and objectively. Speed and objective ratings seem to be two common goals in the research. We chose this article because we wanted to find research examples where text categorization techniques were applied to a semantic network. Their attempts to categorize student reading comprehension relate to our goal of categorizing sentiment. This text also introduced an ontology, and "semantic annotations" link text fragments to the ontology, which we found to be common in semantic text analysis.