Representing Textual Passages as Graphs to Support Question Answering

by

Tyler Christian

A thesis presented to the Honors College of Middle Tennessee State University in partial fulfillment of the requirements for graduation from the University Honors College

Fall 2020

Representing Textual Passages as Graphs to Support Question Answering

by

Tyler Christian

APPROVED:

I dedicate this thesis to my father, Keith, and my grandparents, Jane and John, for always reminding me that I can do anything.

Acknowledgements

I would like to thank my family for their eternal support and giving me everything I have needed to do great things in life. I would like to sincerely thank Dr. Barbosa for working with and guiding me through this fulfilling experience. Also, I would like to thank the Honors College for presenting the opportunity to be involved with them and the people there who have worked with me.

Abstract

As technology and its capabilities are profoundly increasing, the means of communication between humans has become immensely easier. Technological advances are ever more applicable with Artificial Intelligence and the processing of natural language data. However, a problem that is worth pursuing is the concept that machines processing natural language data have little comprehension during question answering. Machines often have difficulty understanding information because data are usually not represented in a comprehensible manner. To take a step toward solving this problem, this thesis will explore a new, automated way to represent any short news passage in the form of a graph. Such graphs are useful because they represent the most amount of information while being compact and leading to accurate, efficient answers. The ability to see relationships throughout entire passages, having properties that make question answering possible, and being able to graph any short news passage bring immense value to this project. This project carries significance because of the fact that it is interfaceable with other systems, simplifying work by serving as a driver and being able to be combined with additional tools. The Natural Language Tool Kit, Neo4j Database Software, Stanford Core Natural Language Processing, and the Python programming language are all tools that were used in completing this project.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

The ultimate goal in the field of Natural Language Processing is to grant machines powered by Artificial Intelligence the ability to understand and interact with humans in everyday conversation. In other words, interactions from human to human should be replicated from human to machine or even machine to machine. Replication entails thinking, processing patterns of speech, and, in this project, engaging in conversation through question answering based on machine comprehension. Currently, there are already significant, impactful applications of this field. Examples include Apple's Siri, Amazon's Alexa, predictive text, and language translation. While the uses of these applications are diverse in everyday scenarios, most of these tools do not explicitly represent information that can be reasoned through in a clear, concise manner.

Of course, applications are created in order to benefit the user, and one such benefit is convenience. Therefore, it is important to explore avenues of automated applications. Automated applications are efficient when neither the user nor the programmer needs to manually input what task is needed to be done. In addition, automated processes are almost always created so that the program or machine is able to process any data generically and handle the largest number of cases possible.

The goal of this thesis is to facilitate a new way of representing short news passages, gathered from CNN, in the form of a graph, in the attempt to improve tasks such as question and answering for Artificial Intelligence (AI) / Natural Language Processing (NLP). The news stories come from the Stanford CoQA dataset (Weston et al. 2015). The full versions of the dataset stories utilized in this study, and their individual sentences evaluated, are listed in the Appendix. During the process of offering this new insight, some issues and questions explored concern knowledge representation, possessing question answering traits, and relationships. Each dataset contains one story and roughly a dozen questions about the story. It is difficult to complete this project if concepts such as part-of-speech (POS) tagging and named-entity-recognition (NER) are the only processes used since there is no way these alone represent textual knowledge in a computer and explore relationships throughout the story. POS tagging refers to categorizing individual words based on their part of speech and context while NER is the process of locating words and sorting them in categories such as locations, monetary values, people, and more. There would also be no representation of various items in the stories, which results in ambiguity. This is where graphs come in.

A graph is a concise representation of data, and all graphs contain a set of vertices (nodes) that are connected through various edges (links). Normally, graphs are used to represent large amounts of information, but in this project, graphs are small in order to maximize representation. Graphs also have the ability to capture relationships. Throughout a data set, these graphs can capture relationships on a small level as well as a large level, and these relationships are inferred based on how vertices are linked, which are through edges. Also, a unique trait of these kinds of graphs is how far a relationship can reach. Instead of connecting vertices that both appear early in the data set, one vertex in the beginning and one vertex in the end can be connected by an edge in the middle, forming insightful relationships.

Each graph lays out an entire passage and is constructed through each sentence containing its own path and types of connections. Since the graphs are small, sentences can also be connected to each other, maximizing representation and locality while

minimizing ambiguity. Graphs consist of nodes (nouns) and are connected by links, which are verbs, prepositions, or a combination of both. Each graph was constructed using the output of a process known as Dependency Parsing (DP) in order to represent words/entities and relationships between words/entities. Nodes will be nouns, adjectives, and some adverbs. Multiple references to the same entity (through pronouns, for example) will be collapsed into a single entity, and each node has its named entity stored as the node type. The relationships connecting the nodes will be one of two types:
1) grammar attributes, such as modifiers to, or 2) sequences of verbs (went), prepositions (in), or both verbs and prepositions combined (appeared_to_glow). This way, the passage is converted from a textual format to a graph format. Utilizing search and graph locality can lead to efficient searching when answering questions. Construction of this module will make use of Python, Neo4j graph database, the Stanford CoreNLP tool, and the Natural Language Tool Kit (NLTK). Using this method, the thesis will offer an automated process for graph representation of passages in a manner that improves search accuracy. It is stressed that the scope of this thesis is representation and possessing question-answering properties, and applications that use the graphs created by this method are not in scope. The background of this thesis will be explored in chapter two, followed by the project's methodology, succeeded by conduct, an analysis, and to conclude how the project could be enhanced in future works.

Chapter 2: Background

Possessing an ability to interact with machines in terms of human languages is an extraordinarily powerful capability. The means of doing so is a specialized area in Computer Science that is combined with Linguistics termed "Natural Language Processing" (NLP). This subfield provides methods to process conglomerates of natural language data that can aid with semantic determination and machine comprehension. Despite this technology, there are numerous challenges in Natural Language Processing.

Because humans can communicate with an understanding of the various connotations that certain words carry, teaching computers semantics is a significant challenge: For example, how will the machine know if one is using the word "great" in a positive or negative way? The lack of robustness is arguably the biggest challenge to seamless NLP (Li 2018; Maxwell and Schafer 2010), since humans also have to teach computers what it *means* for a word or language to be used the way it is. Since humans possess intricate methods of communicating, and with increasingly modern technology, NLP is important in aspects of breaking language barriers, Information Retrieval (IR), filtering out unwanted or harmful information, and the ability to process large amounts of data quickly. The time and space complexity of processing natural language data can be overwhelming, especially in regard to cost, accuracy, and efficiency.

For the programmer, one avenue for advancement in NLP research is the answer to the question, how can one improve machine comprehension so that it will represent details in a manner amenable to retrieving facts? Representing lots of information in a compact form with question answering capabilities makes this problem difficult. Simply building models to represent information is not enough because other aspects such as capturing relationships, having question-answering properties, and the importance of compact representation are all factors that need to be considered in this problem. This thesis takes a step toward solving this problem by aiming to represent the information in a news story in compact form. One intuitive way to represent text is through the use of queryable graphs. In these graphs, every non-verb word of a passage or text is represented as a node and in turn is linked through verbs, prepositions, or a combination of these.

There are decent numbers of methods in which a graph of this category can be constructed, with examples being the combinations of knowledge bases and entity-linked texts or representing entities as low dimensional vectors (Huang et al. 2019; Weston et al. 2015). To phrase this differently, although numbers of methods exist to create these types of graphs, the applications of these methods overlap. This is explained in the demonstration of analyzing two graphs with the same functionality. For example, one graph could be constructed using Open-Domain Question Answering (QA) (which concerns questions about nearly anything), and the other could use Question Answering over Knowledge Graphs (QA-KG) (using facts in the knowledge graph to answer questions) (Huang et al. 2019; Weston et al. 2015). Despite both graphs being different in terms of attributes and construction, the concept of representing knowledge for question answering is used in both. In the end, this simply means that numerous graphs in this domain are possible for representing text for question answering.

No matter the method used, verbs tend to capture how objects and subjects in sentences are tied together. Consequently, if verbs and their semantics are so connected, then the machine is likely to have a better representation of the meaning of a sentence.

Because NLP is a field with numerous applications, the boundaries of this project must be explicit and specific, which is why the scope of the project is limited to producing such graphs. Thus, applications that use the output of these methods are not in the scope of this work.

Upon completion, the proposed graphs will illustrate and offer insight regarding efficient representation through parsing, capturing relationships, and creating links to various parts of sentences, such that a machine would “understand.” To be more specific, these graphs will be able to capture a short news story and its contexts: Therefore, this thesis will improve machine understanding by offering a compact way to represent information and its attributes from these stories. In addition to attempts to find these solutions, aid will also be delivered in improving compatibility by becoming interfaceable with other systems, simplifying work.

2.1 Related Works

Natural Language Processing (NLP) is becoming increasingly important, especially since human communication is immensely complex along with the rise of large amounts of natural language data and text to process. Ray et al. (2018) further discuss difficulties in NLP in a study that more or less “introduces” the challenges of NLP by discussing how processing Big Data is difficult and, most importantly, why information is difficult to represent. Positive correlations with this study’s main discoveries reside within Maxwell and Schafer’s (2010) study, but there are more specifications in Information Retrieval (IR), whereas in Ray et al. (2018) the issues discussed are more general, signifying how NLP’s challenges impact a range of disciplines.

There are also others who have explored various avenues in creating graph representations from text. While their overarching goal has similar traits with this thesis, the graphs themselves are different in terms of appearance, variation, and the process by which text is graphed and represented. For example, Hassan, Mihalcea, and Banea (2007) also aimed to represent passages in the form of a graph but did so by estimating term weights and co-occurrence to measure a dependency's importance. This method used a random-walk model to create graphs and capture how and how much certain words contribute to contexts. Hassan, Mihalcea, and Banea's work is different from this thesis in that this project does not use weights in a random-walk model. Instead, this project explores sequentially graphing stories in a compact form to capture relationships by DP, not by weight and contextual significance. Puente et al. (2013) also implemented graph representation. While these graphs are also different than those of this thesis, there are more similarities. By far the biggest similarity of Puente et al.'s graphs and this project's graphs lie in reduction by collapsing, except that synonyms were used as opposed to coreference resolution, on which this thesis also focuses. These graphs are also termed to be casual graphs while this project creates graphs depicted from factoids (knowledge representation). From both mentioned works, it is inferred both that graph representation is not new and that there is no one true method or answer. This thesis and other works focus on creating graphs to capture relationships and context, but the means of doing so is different.

Chapter 3: Methodology

The problem in this thesis was one that, in the end, had no concrete, fixed solution. This is because there were numerous special cases and limitations. Before illustrating how this process was done, it is important to be prompt and direct about the limitations and scope of the project. First, applications of this thesis and its queries are outside of the project's scope. In this process, it is stressed that the sole task is to take a short news passage story and automatically create a compact, yet representative, graph with properties that makes question answering possible. On a global level, subtitles are not processed, as they have little semantic meaning. Last, no solution will be completely absent from the unavoidable, omniscient special cases. Each step in the development of this automated representation has its own limitations or concepts that will subsequently be examined.

Each story is graphed sequentially. To graph such stories, there need to be methods that gather what is necessary to connect, so that later on proper relationships are captured. Nodes are by far the most important resource for these connections since they serve as the beginning and end of relationships and often centralize in the middle of the graph to give a view of the main person or idea of the story. Nodes are, for the most part, connected by verbs and prepositions, which are between noun nodes, so all non-noun tokens between two nouns in a sentence are joined together to form one link, which connects nodes. After gathering what is needed to make connections, any relationship that is non-sequential is connected right away. A relationship is non-sequential if information is immediately available without reliance on a token's dependency parse.

Dependency parsing refers to analyzing how a sentence is grammatically structured. In a sentence, certain words are deemed as the source of the dependency parse, and these words modify the destinations of a dependency parse. Once the source is linked to its destination, a relationship within the sentence is formed, describing how the source modified the destination. An example of a non-sequential connection could be if a sentence involves a possessor, the reader immediately knows who possesses what and will not have to identify relationships within the sentence or story to comprehend what has taken place. Similar situations arise for modifiers since no iteration is needed to gather information about the modifying and modified. One example of a modifier could be describing the type of prosecution being carried out against a person. Instead of just mentioning "prosecution," a passage could further describe what kind of prosecution is being carried out ("U.S. prosecution"). Now, if there were a question asking what kind of prosecution was being carried out, the answer would be "U.S. prosecution." "U.S." is therefore a modifier to "prosecution" since it specifies what kind of prosecution was carried out, giving further details and making a unique relationship. After immediately connecting what is possible, the rest of the sentence is sewn together through the sequential links by going through the entire sentence from start to end. Sequentially connecting the sentence is the most important step since it is where most of comprehension comes from. Last, the final node of a sentence is connected to a period if nothing separates the last node and the period. It is at this moment that an entire sentence is connected, so the process iterates to the next sentence, where the exact same steps are executed through the story's end.

One last global element is the concept of an externally implemented coreference resolution. Coreference resolution is the process of specifying pronouns by tying a pronoun to a particular entity. As an example, when a pronoun is coreferenced, the person's name takes the place of the pronoun, which helps with collapsing multiple entities in one. So instead of making nodes for both nouns and pronouns, coreference resolution allows the collapsing of pronouns into previously created nodes, resulting in fewer nodes. Since coreference resolution is *external* to this project, its limitations are not addressed within the scope of this work. One of the limitations encountered is in names from other cultures. In a generic western name, the coreference lies within the person's last name; however, names from different cultures can have the coreferencing within the person's middle name. This difference creates issues because in text, any sequential references to people are their last name, and never to their middle name. Because a node's full entity lies in a person's last name, referring to a person's middle name creates inaccuracy, resulting in destinations or sources of relationships to be nonexistent nodes. This limitation can also result in the creation of two nodes for a person with a name from a different culture: one for the full name and the other for only the middle name.

3.1 Gathering Nodes

To sequentially connect one passage, individual words in each sentence need to be processed and examined based on POS Tag and DP in order to identify and establish relationships. Before examining relationships, a beginning and end of one connection need to be established. This is where the first step comes in: finding, identifying, and creating all nodes in the story. A start, end, and period (.) node is generated for each story so that each sentence has a central, unified path at both the beginning and end, with unique paths in between. After the start and end nodes are placed, the story is iterated through one sentence at a time. Since all nodes are nouns, each token of a sentence needs to be checked to see if its part of speech is a noun. If so, a node is created and established based on two cases: 1) if a node's NE is not present, then the node's identity, or the name of the node, is simply the token itself, or 2) if a NE exists, the full recognition is parsed and the second half of the NE is used to name the node. After a node is created, it is added to a list so that, in future sentences, nodes will be made if, and only if, the token is not in this list. An important part of this compact approach is having zero duplicates, so each time a token is found to be a noun, it is checked to see if it is in the list of tokens that were already created as nodes. If it is, the token is simply dismissed so that another node of the same name is not created.

3.2 Sequential Links

Nodes are only half of a relationship, so the next step would be to gather all sequential links in the sentence so that, later on, the sentence can be sequentially connected. All sequential links are made of verbs, prepositions, or a combination of these and are always between two nouns. In order to find these, the sentence is iterated again. Upon encountering the first verb or preposition (i.e., "appeared"), spots are looked at ahead, and every token ahead that is a verb or preposition is combined into one link (i.e., "appeared_to_dim"). Once a token that would be a node is encountered, the link is packaged into a dictionary with the key being the first word ("appeared") and the value being the entire link ("appeared_to_dim"). Dictionaries are a data structure in which a key is used to index through, and when the indexed key is a match, the value is retrieved. In this case, the values are all the combined verbs and prepositions, which are needed for the connection. If a future key ends up being identical to a pre-existing key, the key is extended by another token so that all keys are unique (“appeared_to”), and the values can either be unique or the same since the values are not used to index (“appeared_to_glow”).

Sometimes, duplicate keys arise using this method. Should duplicate keys occur, one additional spot is looked ahead, which results in the key being two words (“have” being one key and “have_denied” being a later key). Even though verbs connect two nodes, not all links with verbs are sequential. Verbs and prepositions also connect subjects, and these relationships are non-sequential because a reader can immediately identify who or what the subject of a verb is. Therefore, when iterating through a sentence, the token’s DP type needs to be checked, and if the type is related to a subject, a flag is raised to indicate that the following verbs/prepositions connect a subject. If the flag is indeed raised, then the first word and the entire link are stored into a separate dictionary. This second dictionary contains all keys and values for connecting subjects and items are only added if the flag is raised.

3.3 Various Cases of Non-sequential Connections

As mentioned earlier, non-sequential connections are relationships that are immediately connected, and a relationship is considered non-sequential if the reader is immediately able to draw the relationship. Throughout this project’s development, there have been varieties of relationships that are non-sequential. In general, these types of relationships are connected by using a token’s DP to locate the source and destination, which are then connected depending on what case it fits (modifier uses “mod” to connect, possessor uses “poss” to connect, etc.). Non-sequential connections are made as soon as possible in order to reduce the number of connections to make later on in the process. Another benefit is that it makes these later connections all one type: sequential connections. So, the general goal is to locate and connect what is possible immediately and to sequentially connect the rest of the sentence later. From these connections, ambiguity is less of a worry and focus is shifted into the more important sequential connections.

Subjects are an ideal case to start with since the concepts of connecting subjects have already been introduced. Essentially, subjects create relationships in such a way that a sentence’s path has a “shortcut” due to subjective relationships being direct. The following figure shows a sentence in graphed form, whose path has a shortcut created by a NSUBJ connection.

"We don't know who has come up with this huge lie," Choquehuanca said, adding, "We would like to let the international community know that the rights of aerial traffic for Bolivia have been violated."

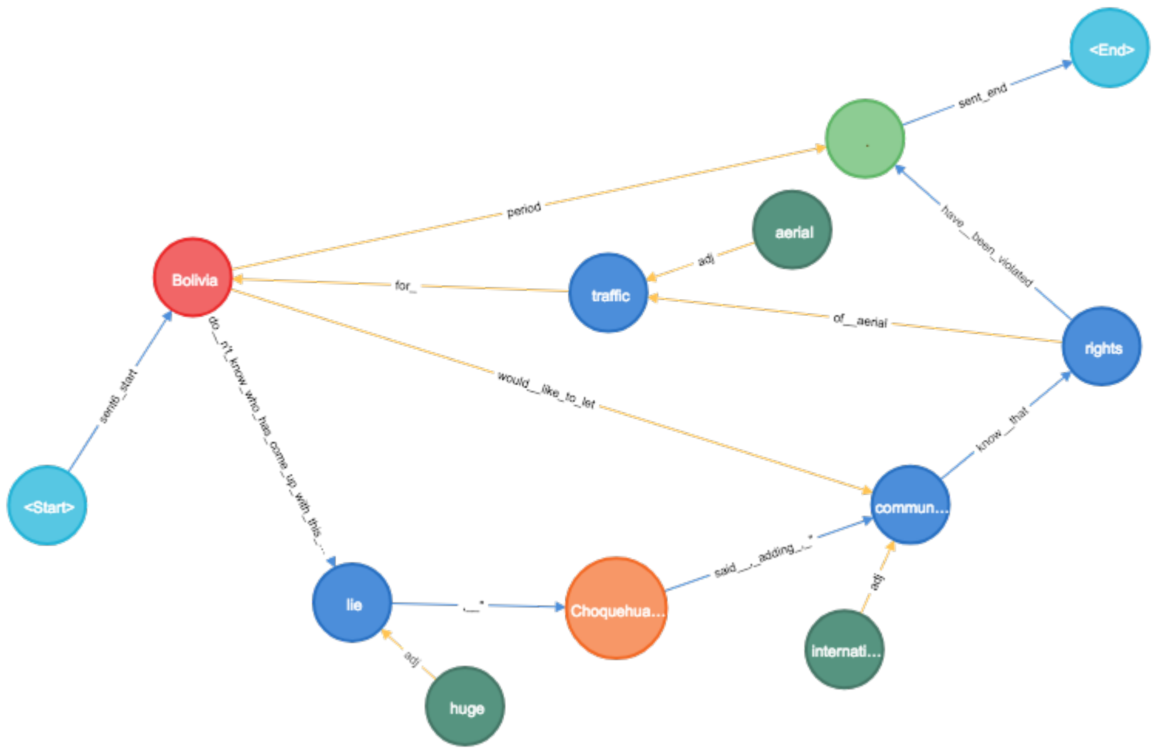

Figure 1: One fully graphed sentence with shortcuts (blue links) in its path due to having a subject connection.

To detect a subject relationship, each token's DP type is checked, and if the type is "NSUBJ" (representing the nominal subject), that token's DP source and destination are located. Next, the token after the source of the DP becomes the key and is checked against the dictionary of subjects for a match. If there is not a match, the rest of the sentence is examined and checked against a dictionary to find a match. From there, a relationship is captured by connecting the source of the subject to the destination (the object), with the value of this matched key to bridge the two together. This particular case is the most difficult in terms of establishing connections because the words that connect the nodes are always different (verbs and prepositions connect subjects, making the connection unique). Furthermore, there are also cases where the key does not exist (one spot after the source of the DP), so more processes are carried out, depending on the case, to ensure that there is always a match.

Further complications arise because these connections make two of the same kind. A non-sequential subjective connection and a sequential connection using the same verbs are made, which takes away from the overall concept of minimization. The following figure depicts one graphed sentence having duplicate subject connections.

"Three—Brazil, India, and Poland—have denied the request outright."

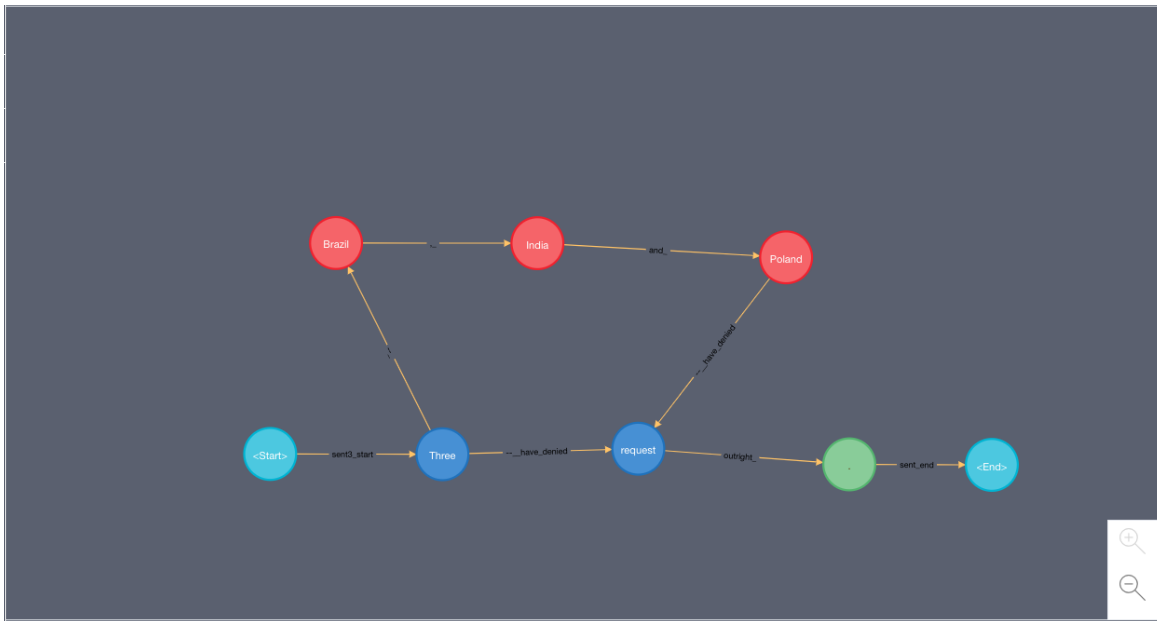

Figure 2: A fully connected sentence with two identical relationships due to one concerning a subject - a non-sequential connection.

To combat this, another dictionary is made. This secondary dictionary is parallel to the dictionary or verbs, and it bookkeeps whether or not a connection using a set group of verbs of prepositions have been already made. Parallel dictionaries are the same size as their counterparts, and the keys are identical to the primary dictionary. Upon encountering a relationship whose type of DP concerns subjects, a key is found and a connection is made using the key's value, which is the actual group of words. After the connection is made, that same key is used to change the value in the second dictionary to "True" in order to indicate that the connection has already been made. Bookkeeping using these dictionaries is important because when the rest of the sentence is sequentially connected, if the connection has already been made, another is not made to maintain minimization. For example, in the sentence *"Three—Brazil, India, and Poland—have denied the request outright,"* "Three" is the subject and is connected to "request," which is the object, by the link "have_denied." Of course, this link contains verbs and prepositions, as do all links for sequential connections. Because this link has already been used, it is essentially turned off by setting the value in the parallel dictionary to "True." By changing this value, the verbs and prepositions are skipped when connecting everything else sequentially.

A second kind of non-sequential connection is modifiers, where "nmod" connections bridge a modifying and a modified together. Generally, connections of this convention are made when a token has a "nmod" or "compound" type of DP. The source of the DP is termed as the modifying and the destination is deemed the modified, both of which are nouns unless the relationship concerns adjectival modifiers. Modifiers are easy to establish with the exception of people. According to Stanford NLP, a person's last name modifies his or her first name, and since the node type contains the person's full name, this type of connection is avoided. Remaining cases can be packaged together. First, adjectives are also another type of non-sequential connection, and as long as the adjective is not an adverb, the connection is made. Since all nodes prior to this point have been nouns, more nodes need to be made, which are the adjectives themselves. Once the adjective node is created, it is connected to the destination of the DP, a noun. Possessors are the last case and are simple as well. Upon encountering a possessor, the source and destination are determined and the connection is created. If either of the two is a pronoun, the coreference mapping is used.

3.4 Sequential Connections

After generating nodes, links, and connecting what is possible immediately, the remainder of the sentence is connected sequentially by using tokens' DP as the path, resulting in sequential connections being the bulk of representation. The sentence is iterated through once again until the first noun or pronoun is detected. Upon detection, this token will serve as the beginning of the sequential relationship. However, before the second noun is found, there are various cases to examine.

In a sentence, the very first token may not be a noun ("While Bolivia and Venezuela. . ."). Instead, the first token could be a verb or preposition ("While"). Such a case is detected by examining the first token, and if it is not a noun, it is used as the key to match in the verb dictionary. Once a match is found, the parallel Boolean dictionary (one that holds True or False values) is examined, and upon a false value (which means that connection has not already been made), the "Start" node is connected to the very first noun of the sentence in addition to the words before the first noun. So, the start node would connect to "Bolivia" with "sent_start" and the start node would also connect to Bolivia with "While" (since "While" is before "Bolivia," the first noun). The rest of the operations for the iteration are skipped. Another case can arise where difficulty is involved, and it pertains to when there are two nouns, one right after the other (". . .on espionage charges. . ."). This is a challenge since finishing the sentence with sequential connections requires a "bridge" of verbs and prepositions in order to connect all nodes. In this case, there is no bridge ("espionage charges"). One answer to this problem is simply to look ahead one token upon detecting a noun, and if one token past the detected noun is yet another noun, then two spots in the sentence are skipped. This is to ensure that the second noun of the back-to-back nodes will serve as the true starting point. Another answer is looking to see if the two nouns form a non-sequential connection. If the two nouns do form a non-sequential connection ("espionage" modifies "charges"), the end of the non-sequential connection is used as the starting point ("charges"). Ultimately, this does not cover every case since some back-to-back nodes are non-sequential, such as a modifying and a modified. These additional cases create further complications, so the aforementioned ban list serves yet another purpose, which is to minimize the amount of back-to-back nodes. When a noun is detected, it becomes the starting point of the connection.

A sequential relationship has a beginning node ("way"), a sequential link ("to_travel_to"), and an end node ("territory"). Detecting and establishing the first node of the sequential connection ("way") is only one of three parts of creating a sequential relationship. To move forward, sequential verbs and prepositions used for connection are to be sought after. This is done by marking the beginning node's location in the sentence and iterating one spot ahead. By looking ahead one spot ("to"), that verb or preposition is able to be extracted from the sentence to serve as the key to look for in the dictionary of verbs and prepositions, with its value being the full link ("to_travel_to"). Later, when this key is checked against the dictionary after finding the second node, there will always be a match since this method is identical to handling duplicate keys when first creating the dictionary for verbs and prepositions. While this method covers most cases, it fails to cover all situations; therefore, if the newly extracted key happens to be a duplicate, yet another spot is looked ahead and is combined with the first spot to form a new key (the key is now two words). A flag is raised afterward to indicate which key to use.

Finding an existent key to check against the sequential dictionaries is the second-to-last step in creating a full, sequential connection. Completing such a process now only requires finding the second node, which will serve as the end of the connection. Looking for a second node involves looking ahead from the first noun (“way”) until another noun is found (“territory”). While looking ahead, it is important to not exceed the end of the sentence since some cases could involve not even having a second noun (i.e., being at the end of a sentence). Should this happen, the period also serves as a noun. More importantly, having a period as the second node would mean that the entire sentence is already connected since verbs and prepositions exist between the last node and a period. In other words, having a period as the second node allows one sentence to be entirely connected from to start to finish, which means that having a period right after a noun leaves empty space. To ensure that this empty space is voided, a later, special connection is used to “finish” the sentence once it is sequentially connected. Until then, other nouns are discovered and are mapped appropriately.

Simple detection of another noun is not quite enough to be able to connect. These endpoints are also sequential nouns, so the newly discovered noun cannot be in the ban list. Meeting these criteria signals that another sequential noun has been revealed, so the connection is finally ready to be made. Coreference mapping is used in the event this second noun is actually a pronoun and the latter part of the token’s NE is upon detection. It is at this point when the beginning (“way”) and end (“territory”) of the connection are to be joined, so the previously created key (“to”) is used to retrieve its value (“to_travel_to”), which are the actual verbs and prepositions of the sentence. Differentiating which key is to be used is based on the aforementioned flag, resulting in a system where a match is always found. Now, as long as the connection has not already been made (by using the same key to check against the Boolean dictionary), a brand new relationship is established with a unique distinction of being sequential ("way" is connected to "territory" by "to_travel_to," which is the value from the key "to"). This entire process is restarted as iteration progresses.

Once all sequential relationships are created, one last step is carried out if a noun is immediately followed by a period, which is to "bridge the gap." To bridge, few iterations are performed starting from the period, and upon detecting a noun, the last noun and period are connected via a "period" connection, which happens to be the special connection previously mentioned. At long last, one entire sentence is fully connected, with the sequential pieces of information forming a path by using tokens' DP, and the remaining sentences of the passage are connected in the same way.

"Edward Snowden's hopes of finding asylum from U.S. prosecution on espionage charges appeared to dim Tuesday as country after country denied his request or said he would have to find a way to travel to their territory to apply."

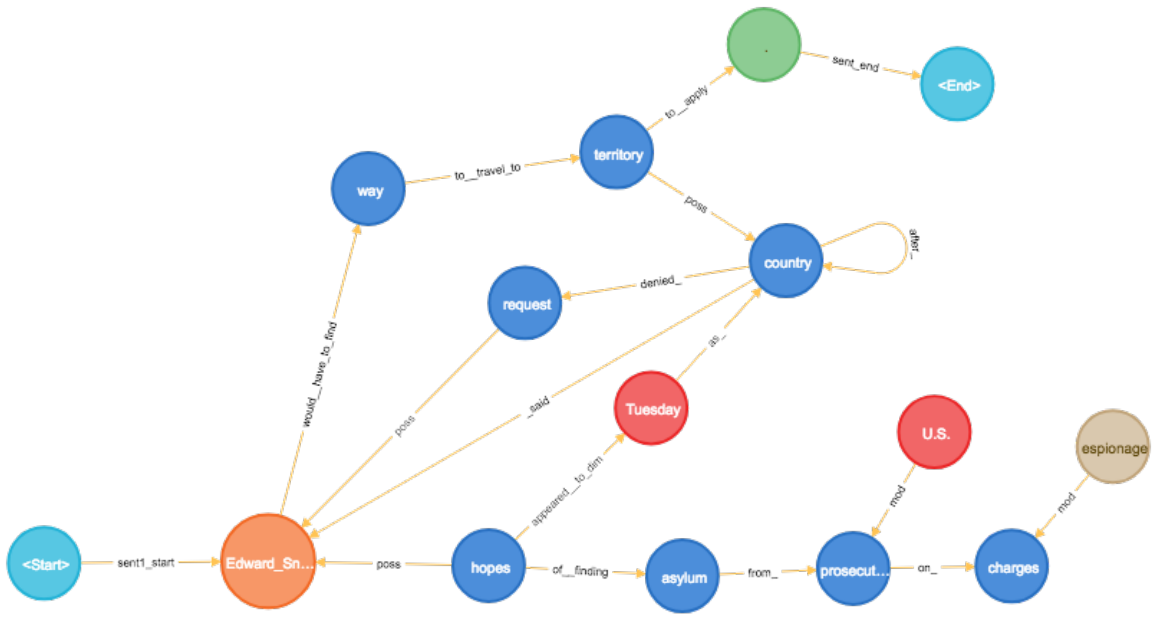

Figure 3: One fully connected sentence.

Earlier sentences take longer to connect since all nodes are generated early on, but later sentences are quickly connected due to the collapsing of nodes. Numerous links to nodes result, allowing relationships across entire passages to be seen as opposed to relationships within one sentence. Another result is sentences having paths that cross, granting the ability to examine relationships between entire sentences. Once every sentence is connected, a compact graph of an entire story is created. Relationships across entire stories are able to be seen, and it is easy to depict who or what is important based on how many nodes and links surround and connect to an entity. Such graphs are also minimized, creating efficiency when interfaced with other systems, simplifying work. No ambiguity is also a benefit since coreferencing is used to map out who or what exactly a later reference leads to, additionally leading to high accuracy. Ultimately, a graph using this method is concise, well represented, free of ambiguity, and accurate.

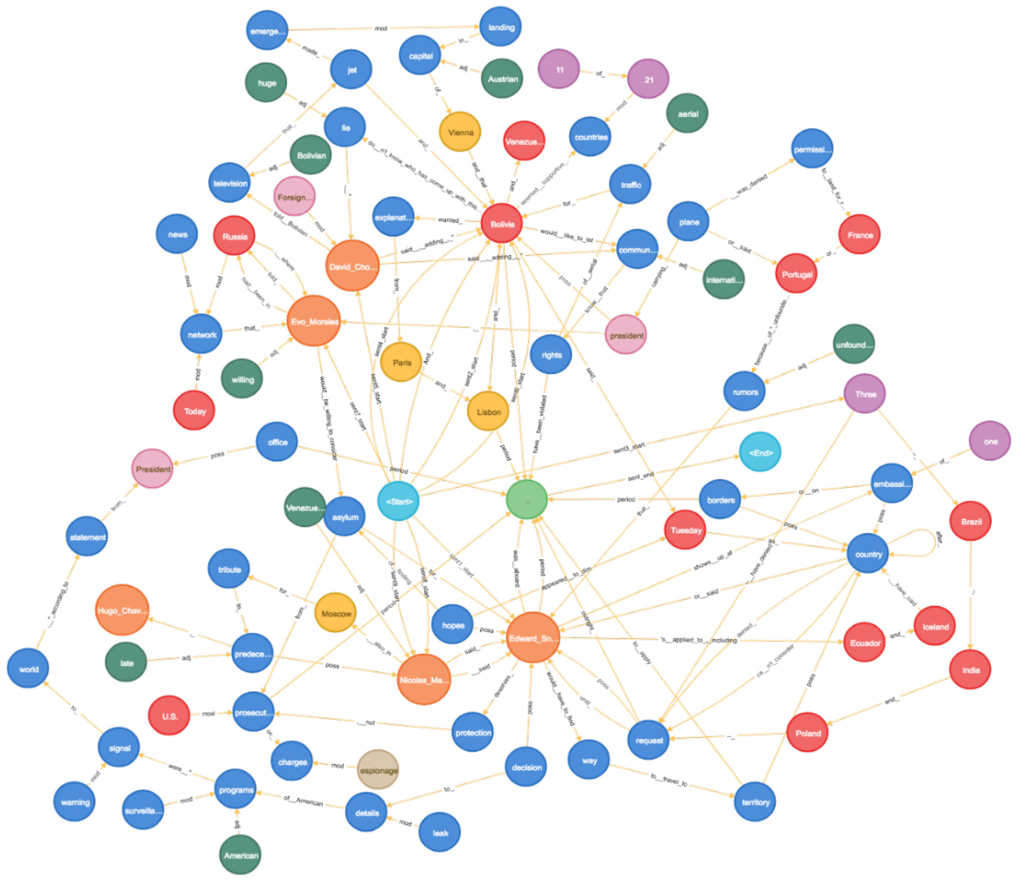

Figure 4: An entire story represented in the form of a graph. Numerous references to entities are collapsed.

Chapter 4: Conduct

Most experimentation was performed with the importance of ensuring this project was able to work with more than just one story while not using logic, inferences, or real-world knowledge since all these concepts lie beyond the scope of the present project. Having a working graph with only one story is not enough to be able to propose that this project offers an insightful, useful answer because one story does not cover anywhere near all cases in text. Creating and representing one story and using the same code to fully represent another story were the remedy of establishing usefulness. Experimentation started with graphing any one story to gauge this method's viability, while the second story was one of more complexity and length where this new story addresses the project's limitations. This second story is needed since it is different in numerous ways from the first story, and it was more difficult to implement because that meant all written code had to cover every case from different cultures or environments across both passages. At first, while the original story was able to be graphed, the second was not, so more experimentation was needed to modify code for the second story without breaking the first.

Because of the importance of being able to graph multiple stories, most challenges of experimentation lay in ensuring that all code written was generic, which meant that the project needed to be able to load at least two distinct stories and to process and graph them with no complications. This overarching challenge had individual challenges inside, such as having a way to address project limitations while working with both stories and ensuring all cases within both stories were covered with one script of code. Addressing project limitations across both stories was the most difficult challenge, and since the solutions to these were beyond scope, manual interference was needed. This simply involved "turning off" the manually implemented coreference resolution (depending on the story) and code that deliberately dismissed subtitles when switching between stories. Such a challenge would be fully solved when these limitations were addressed in full and if no manual interference was needed when changing stories. Other challenges were able to have full solutions. One of these other challenges included ensuring that the "Start" node connected to the correct node upon a sentence's beginning. For example, in the second story, a sentence started with "*A Sudanese woman has been freed . . .,"* which meant that the "Start" node needed to connect to "woman" instead of "Sudanese." Since "Sundanese" modifies the type of woman, that node is not sequential and is therefore the incorrect node to connect. When implementing the first story, such a case was not considered because it did not occur; but, the case made its first appearance in the second story, so it needed to be addressed. Simply fixing this error for the second story was not enough to say the challenge was solved, since fixes were needed to be made in a way that they did not "break" the first story. This challenge was solved by devising other possible cases that were not addressed previously while also considering the first story's cases. The overall result was a brand-new, restructured section of code that handled all cases of choosing which node is the first sequential node across all stories. Figure 5 shows a non-problematic sentence from the second story and how overcoming the challenge made the sentence correct while not breaking the first story. It also shows that the method used to create graphs works generically—for any story.

"A Sudanese woman has been freed from prison a month after being sentenced to die by hanging for refusing to renounce her Christian faith."

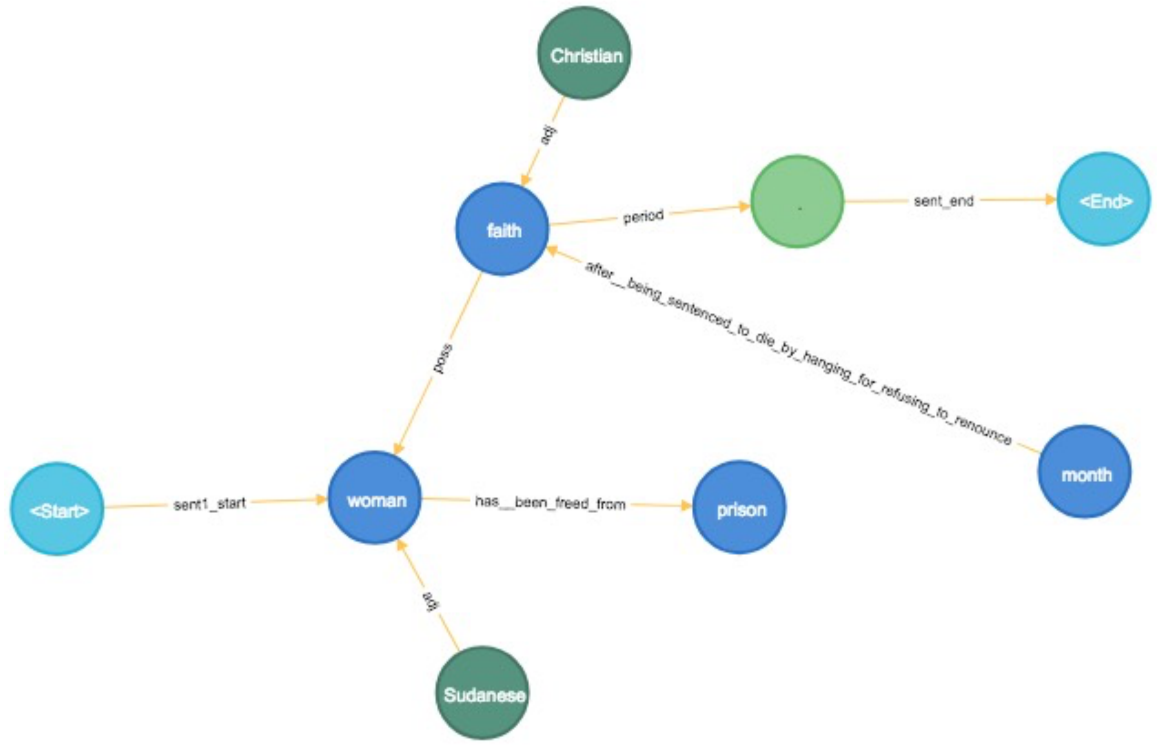

Figure 5: A fully graphed sentence from another story.

Modifying code to work with both stories marks the end of the first phase of experimentation. The next phase of conducting experiments consisted of manual editing in order to address this project's original limitations across all stories while ironing out any special cases. Manual edits include commenting and uncommenting code that took out subtitles depending on which story was being graphed and addressing story-specific special cases (there was only one where manual editing was needed). Because of these manual efforts, further experimentation proved to be equal across both stories in order to gain an analysis.

Further experimentation included methods such as running the project numerous times and averaging the time taken to graph for each story. Another method was grouping and ungrouping nodes based on a wider range of node types. So, instead of storing nodes as only nouns, adjectives, etc., nodes were stored with the node type being NEs such as titles, locations, cities, crimes, and much more. While NER was able to specify types of nodes, it took away the overall idea of graphing *nouns* and also made it more difficult to graph the second story because some NEs were those from other cultures (the second story), which directly clashed with this project's limitations: Therefore, some of the NER method was reverted and further experimentation stored as many node types as nouns as possible. Note that there are indeed some specified NE node types, but these specific NEs are general and did not impact the second story (i.e., people, titles, etc.). Because of the ways these conducts and experiments were set up and performed, a more intricate analysis was able to be formed since the analysis was across both stories, adding even more validity and justifying this work's use in other, larger systems.

Chapter 5: Analysis

Experimentation was able to both show and prove this project's validity by being able to represent not only more than one story, but also stories focused on different cultures and demographics, despite the overarching limitations from coreferencing, subtitles, and more. By far the most significant conclusion is the process of ensuring each graph had question-answering capabilities. Sequentially graphing text while using methods such as DP and connecting subjects gave graphs aspects capable of answering questions. Sequential graphing further enhanced the generic goal of this project since it means that larger systems could implement question answering with multiple, generic questions. Questions with words not matching items in the graph are also able to be asked, and each graph can show how one may answer a specific question. Experiments proved that this project is also viable in being able to collapse sequential links into one, because numerous verbs and prepositions were able to be combined into one link as opposed to having one link for each individual verb or preposition token. Reduction of these conglomerates resulted in minimal graphs, while explicitly establishing graph locality, which in turn reveals who or what is important based on the neighborhood of nodes around an entity.

Reduction being another important takeaway is further justified in being able to capture relationships across the entire stories as opposed to just one sentence. For example, a person can be graphed as a node and have a relationship in the first sentence. Later in the story, the same person could form a new relationship. Because of the drastic reduction in terms of how many nouns and people were graphed, both relationships are drawn to the same node instead of making another node of the same type. The following chart in Table 1 shows how many total nouns a story had versus how many nouns were actually graphed.

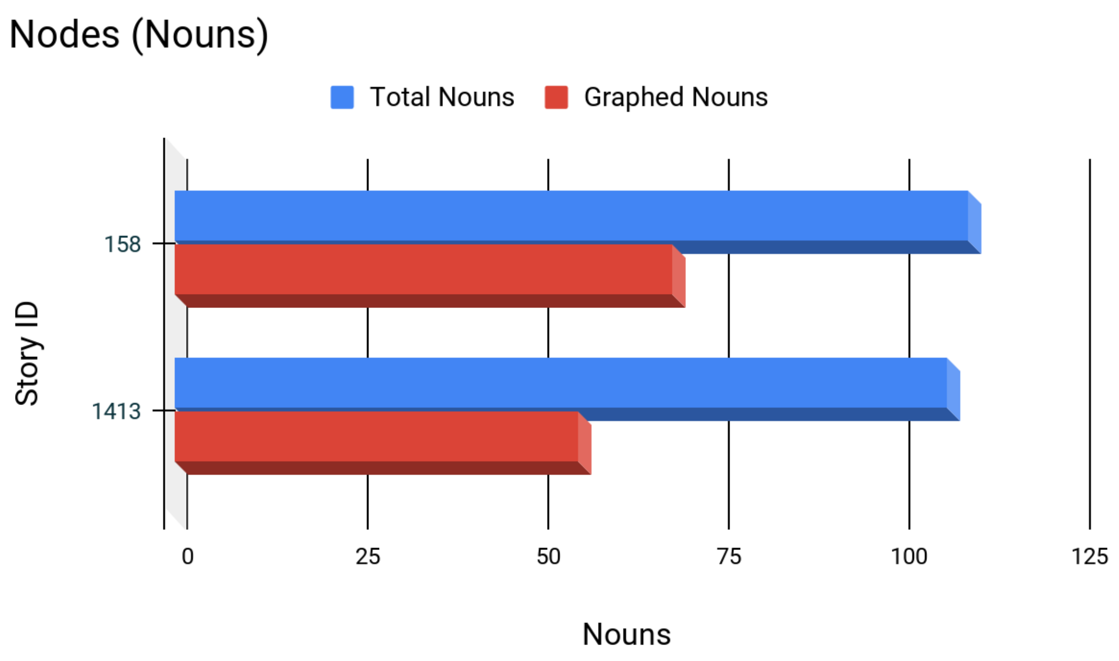

Table 1: The total number of nouns per story versus how many were graphed.

Table 1 shows a consistency of reducing the number of nodes generated by almost half per story. This level of reduction is significant as it shows exactly how much minimization occurred for nouns alone and implies that the total amount of time to create nodes from nouns is reduced by almost half as well. Even greater results are yielded from using larger stories since larger stories require more time and space complexity. Table 2 shows a similar overview of how many verbs and prepositions each story had versus how many links were made.

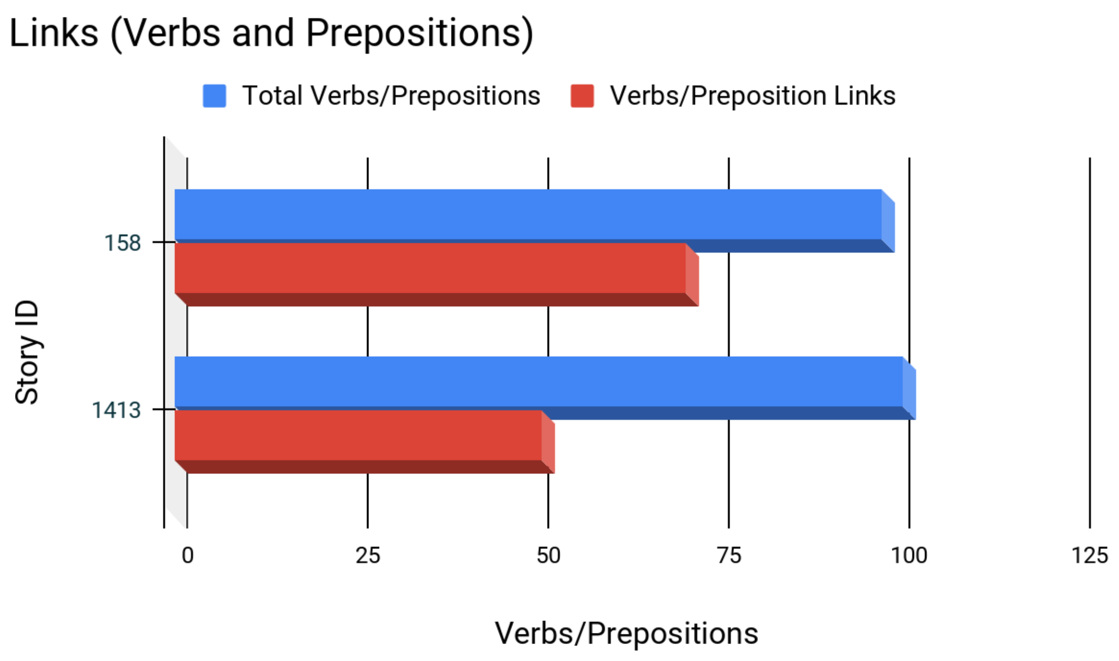

Table 2: The total number of verbs and prepositions in each story versus how many links were made.

Table 2 shows more consistency in terms of how much was reduced. While these consistencies are similar to that of Table 1, Table 2 is different in its correlation: the more total verbs and prepositions a story has, the larger the reduction is made, and the fewer links are graphed. This would mean that if a story of immense size was graphed, the reduction of verbs and prepositions is even larger since a lot more words of these parts of speeches would be packaged into one. Basically, the more verbs and prepositions, the larger the reduction, hence fewer links. At a base level, both tables show reductions of about half, and these are for relatively small stories. If larger stories were graphed, it would be feasible to pose that reduction would be by *more* than half.

Apart from these significant reductions, it is also important to address whether or not a story could be a great or terrible example. For the most part, a story was considered good if there were few back-to-back nodes, numerous verbs and prepositions, and few special cases. Numerous verbs and prepositions create lots of sequential connections, resulting in large numbers of reductions, while few back-to-back nodes create smaller ban lists, handle fewer special cases, and establish fewer relationships when making sequential connections, aiding in performance. On the other hand, a story served as a bad example if it had names and entities from different cultures, few verbs and prepositions (which would lead to few sequential links), and numerous back-to-back nodes (such as entities that were three words long). Names and entities from other cultures meant either further addressing this project's limitations or manual interference to keep the passage in a graphed state. More back-to-back nodes resulted in larger ban lists and more evaluations when connecting later, creating hindrance in speed. Even if a story with numerous limitations was graphed, results in reduction and representation remained the same. Because of this, this project has significant validity in that it is able to be inserted into different kinds of larger systems.

Another form of analysis in this project was its significant validity about where its applications could lie, with emphasis in question-answering modules. This new, automated graphing method can be used in other portions of NLP as well as in fields that are vastly different in their own disciplines. For example, this project could serve as a driver program for a larger system that uses queries to fully implement true question/answering. Question answering is also why this project's intent was more than just "building a graph"—there needed to be properties that make question answering possible. While this project does not actually answer questions (since that lies outside the project's scope), one is still able to see what kinds of questions could be asked and how they may be answered using the graphs produced. A larger system that implements this deems completion. Questions asked may be ones that assess reading comprehensions by attempting to retrieve factoids as answers or ones that do not explicitly match items in the graph. This larger system could use queries as the questioning portion and various methods such as using synonyms or taking lemmas (the base form of a word) would be the answering part. Although the kinds of questions that can be asked are limited (since there is no real-world knowledge implemented in the graphs), the use of synonyms could diversify what kinds of questions can be asked. As of now, one could only look and "see" how questions could be answered, but a future system would make this plausible. Figure 6 shows an example of what a question (Q) may be and how the answers (A) could be retrieved simply from inspection.

Q: "Was something violated?"
A: Yes.

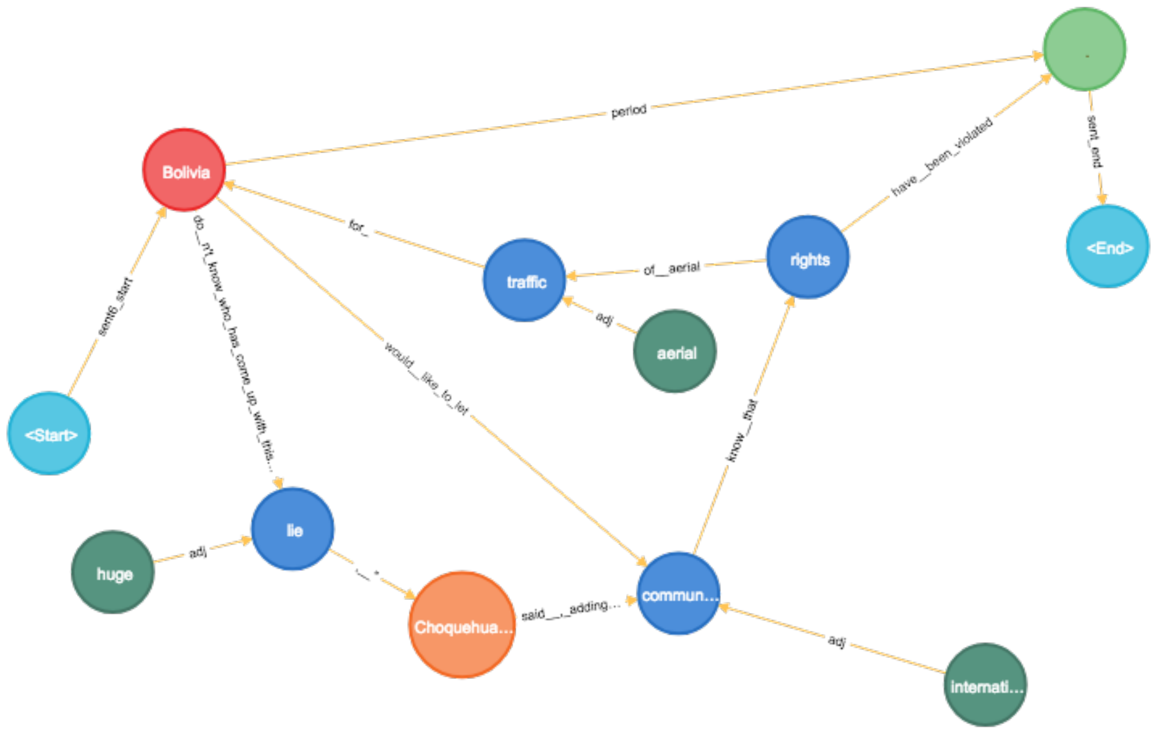

Figure 6: A graphed sentence showing how the answer is retrieved.

By using Figure 6, a larger system could "look" at the graph and see how the question could be answered. In this case, the question would be answered by locating the "have_been_violated" link and unpackaging it into individual tokens, focusing primarily on "violated." After unpackaging, a note would be made that this link came from the node "rights," which would signal that something had been violated. A unique trait of these graphs is the ability to answer questions that are worded differently, meaning multiple, different questions could be asked. Questions could be asked where there are almost no matches between the words in the question and the graph. This could be done by using lemmas. So, another question such as "*Has there been a violation?"* would have the same answer, despite "violation" not being in the graph. By relying on lemmas, the base forms of words are extracted, which would mean multiple versions of a word have the same root. This would mean that both "violated" and "violation" have the same root, "violate." If this method were used for both questions, the answers would be the same despite the questions being unique but similar. Figure 7 shows another example.

Q: "Who said the plane had an emergency landing?"
A: David Choquehuanca.

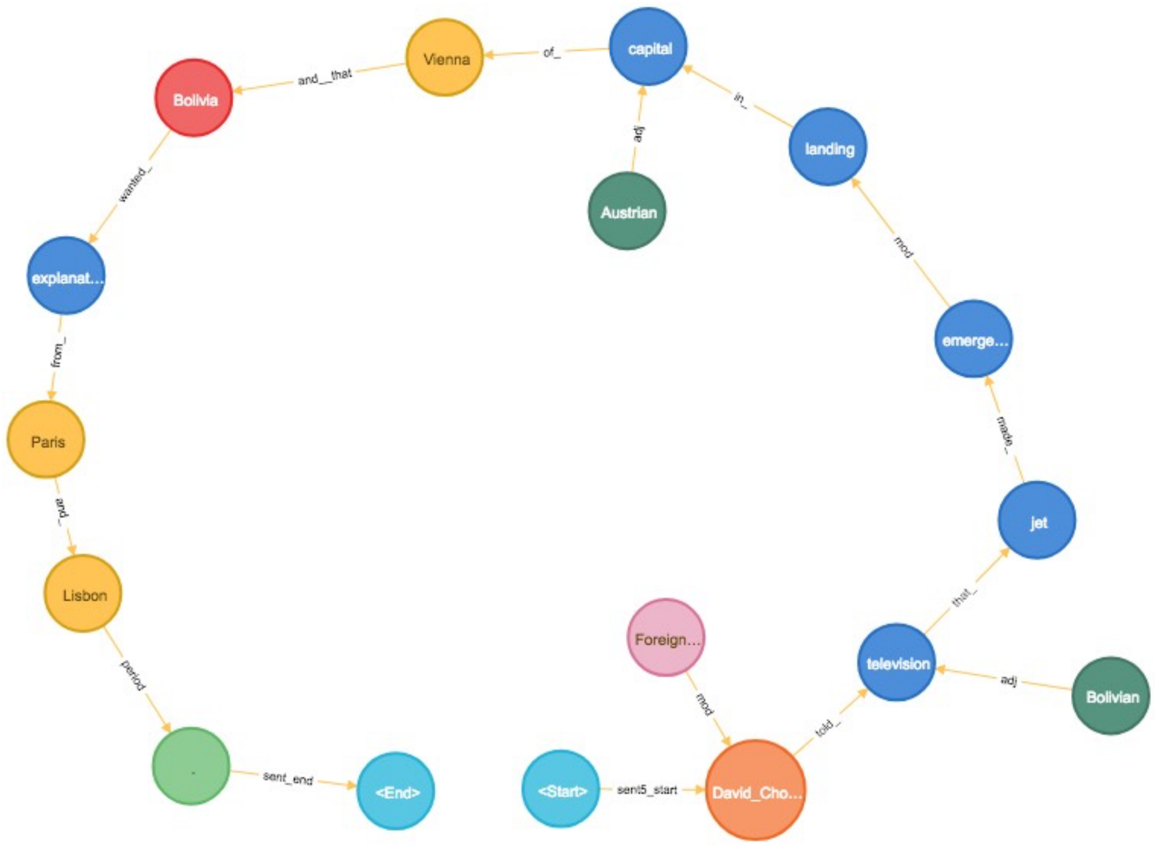

Figure 7: Another graphed sentence that is useful in answering questions.

Figure 7 and its question could be answered by using another method: finding and applying synonyms. Wordings of questions do not have to have an exact match with graphs in order to find answers, as seen in Figure 6 and through using lemmas. Using another method, such as implementing synonyms, is powerful because it allows questions to be asked in different contexts while further enhancing generic use. Locating and unpackaging relevant links remain the same; however, in this specific example, the words "said" and "had" do not match any of the unpackaged links ("made" and "told"), but when a method using synonyms is applied, one would see that "said" is immensely similar to "told" and "had" is similar to "made." Once these similarities are pointed out, it becomes apparent that while the context is different, the intent is the same. This means the same, correct answer is returned despite the question being in different context. In the end, using synonyms allows for more answerable questions despite using words that do not lie in the graph, allowing for the more emphasized generic use.

As a whole, this analysis has shown that this project is successful in creating minimalist graphs that represent the most information. Reductions are of nearly half for smaller stories, and consistencies have been shown that larger stories would produce even larger reductions. Outlines of what consists in great stories have been explored along with how great stories affect processing, while terrible stories and how they hinder this project's goal have also been explored. Furthermore, analyses exposed this project's validity in that it is easily interfaceable with other systems simplifying work in fields within NLP as well as fields that are independent of this project's realm. In the end, these contributing factors have shown that this project is indeed able to automatically convert any passage into the form of a graph, which further shows how these graphs represent the most information while being minimalist, resulting in large reductions and the collapse of information. Graph locality, possessing question-answering capabilities, the ability to capture relationships on a global level, efficiency, and accuracy are all sufficient justification of this project's wide use of applications. While not a complete solution, this project has numerous benefits and is of significant validity, and is a truly viable piece of work.

Chapter 6: Conclusion

This thesis has described the concept of developing a process that automatically converts short news stories into compact graphs in such a way that question answering is improved and more efficient among AI. While there are limitations and ideas that are out of scope of this project, the automated process undertaken in this research is able to graph short news passages, making the project viable in its diversity with what kind of short stories can be represented through a graph. Applications also lie in fields other than Computer Science, such as Linguistics and Education, since the project is also usable as a visual tool.

6.1 Future Work

In every story, it is highly likely that there will be at least one token, node, or DP that is unique and non-conformant to normal means. These are known as special cases, with variations to each. Because of this, it is important to note that the special cases contribute to the fact that there is no concrete, fixed solution to this thesis inquiry. There will always be cases that will not be able to be correctly processed, and these numbers will only increase due to the immense complexity of the English language.

Although this project has shown that it is able to automatically graph many short news passages, opportunities for future work and expansions are available that would make the project even more diverse, compact, and automated. The first opportunity of future work would be an enhancement of current coreference resolution methods, and doing so will result in a performance boost. Inside this implementation, there would need to be a method to handle names and entities of different cultures. This is because the NE of full names from many different cultures lies in the middle name, whereas the NE in a regular, generic English name lies in the last name. When a person or entity is referred to numerous times in text, the last name is always used, and never the middle name. Therefore, if the NE is placed with the middle name, it creates inconsistency and can lead to reference to an incorrect or nonexistent node.

Another limitation that provides opportunity for future work lies in identifying and not processing subtitles. In the database that was used, subtitles and text following it were jointly put together as one “umbrella” sentence. Subtitles are not important to the story as they are not sequential and do not provide enough semantic meaning to be treated as an independent sentence. Thus, in future work, a process for identifying and separating subtitles from a sentence would be a solid avenue to undergo. It is important that subtitles are never processed.

Bibliography

Elhadad M. 2010. Natural Language Processing with Python. *Computational Linguistics* [Internet]. [Cited 2019 Oct 21]; 36(4): 767-771. Available from: 10.1162/coli_r_00022

Hassan S, Mihalcea R, and Banea C. 2007. Random-Walk Term Weighting for Improved Text Classification [Internet]. [Cited 2020 Oct 14]; 242-249. Available from: https://ieeexplore.ieee.org/document/4338355

Huang X, Zhang J, Li D, and Li P. 2019. Knowledge Graph Embedding Based Question Answering [Internet]. [Cited 2020 March 30]; 105-113. Available from: https://doi.org/10.1145/3289600.3290956

Lam S, Chen C, Kim K, Wilson G, Crews JH, and Gerber M. 2019. Optimizing Customer-Agent Interactions with Natural Language Processing and Machine Learning. Institute of Electrical and Electronics Engineers [Internet]. [Cited 2019 Oct 15]; 4010-4015. Available from: 10.1109/SIEDS.2019.8735616

Li H. 2018. Deep Learning for Natural Language Processing: Advantages and Challenges. *National Science Review* [Internet]. [Cited 2019 Oct 21]; 5(1): 24-26. Available from: 10.1093/nsr/nwx110

Liu Y and Zhang M. 2018. Neural Network Methods for Natural Language Processing. MIT Press [Internet]. [Cited 2019 Oct 16]; 44(1): 193-195. Available from: 10.1162/COLI r 00312

Marneffe MC, MacCartney B, and Manning, CD. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. *Proceedings of LREC* [Internet]. [Cited 2020 Jan 20]; 449–454.

Maxwell T and Schafer B. 2010. Natural Language Processing and Query Expansion in Legal Information Retrieval: Challenges and a Response. *International Review of Law, Computers & Technology* [Internet]. [Cited 2019 Oct 21]; 24(1): 63-72. Available from: 10.1080/13600860903570194

Ponti EM, O'Horan H, Berzak Y, Vulic I, Reichart R, Poibeau, T, Shutova E, and Korhonen A. 2019. Modeling Language Variation and Universals: A Survey on Typological Linguistics for Natural Language Processing. Association for Computational Linguistics [Internet]. [Cited 2019 Oct 15]; 45(3): 560-601. Available from: 10.1162/coli_a_00357

Puente C, Garrido E, Olivas JA, and Seisdedos R. 2013. Creating A Natural Language Summary from a Compressed Causal Graph [Internet]. [Cited 2020 Oct 14]; 513-518. Available from: 10.1109/IFSA-NAFIPS.2013.6608453

Pujari R and Goldwasser D. 2019. Using Natural Language Relations between Answer Choices for Machine Comprehension. Association for Computational Linguistics [Internet]. [Cited 2019 Oct 15]; 4010-4015. Available from: 10.18653/v1/N19-1403

Ray J, Johnny O, Trovati M, Sotiriadis S, and Bessis N. 2018. The Rise of Big Data Science: A Survey of Techniques, Methods and Approaches in the Field of Natural Language Processing and Network Theory. *Big Data and Cognitive*

Computing [Internet]. [Cited 2019 Oct 16]; 2(3): 1-18. Available from: https://doi.org/10.3390/bdcc2030022

Reddy S, Chen D, & Manning, CD. 2019. Coqa: A Conversational Question Answering Challenge. *Transactions of the Association for Computational Linguistics*, 7, 249-266.

Sharma LK and Mittal N. 2017. Prominent Feature Extraction for Evidence Gathering in Question Answering. *Journal of Intelligent & Fuzzy Systems* [Internet]. [Cited 2019 Oct 16]; 2923-2932. Available from: 10.3233/JIFS-169235

Sun H, Dhingra B, Zaheer M, Mazaitis K, Salakhutdinov R, and Cohen W. 2018. Open Domain Question Answering Using Early Fusion of Knowledge Bases and Text. School of Computer Science, Carnegie-Mellon University [Internet]. [Cited 2020 March 30]; 1-12. Available from: https://arxiv.org/abs/1809.00782

Weston J, Bordes A, Chopra S, Rush AM, Merrienboer B, Joulin A, and Mikolov T. 2015. Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. *Facebook AI Research* [Internet]. [Cited 2019 Oct 21]; 1-14. Available from: https://arxiv.org/abs/1502.05698v10

Appendix

Abbreviations

AI	Artificial Intelligence
DP	Dependency Parse
IR	Information Retrieval
NER	Named Entity Recognition
NE	Named Entity
NLP	Natural Language Processing
POS Tag	Part-of-Speech Tag
QA	Question Answering

List of Stories/Sentences Analyzed in Study

Story 1

(1.1) Edward Snowden's hopes of finding asylum from U.S. prosecution on espionage charges appeared to dim Tuesday as country after country denied his request or said he would have to find a way to travel to their territory to apply.

(1.2) While Bolivia and Venezuela seemed supportive, 11 of the 21 countries he's applied to, including Ecuador and Iceland, have said they can't consider his request until he shows up at one of their embassies or on their borders.

(1.3) Three—Brazil, India and Poland—have denied the request outright.

(1.4) And Bolivia said Tuesday the plane carrying its president, Evo Morales, was denied permission to land for refueling in either France or Portugal because of "unfounded" rumors that Snowden was aboard.

(1.5) Foreign Minister David Choquehuanca told Bolivian television that the jet made an emergency landing in the Austrian capital of Vienna and that Bolivia wanted an explanation from Paris and Lisbon.

(1.6) "We don't know who has come up with this huge lie," Choquehuanca said, adding, "We would like to let the international community know that the rights of aerial traffic for Bolivia have been violated."

(1.7) Morales had been in Russia, where he told the Russia Today news network that he would be willing to consider asylum for Snowden.

(1.8) And Venezuelan President Nicolas Maduro, also in Moscow for a tribute to his late predecessor, Hugo Chavez, said Snowden deserves protection, not prosecution.

(1.9) Maduro said Snowden's decision to leak detail s of American surveillance programs were "a warning signal to the world," according to statement from the president's office.

Story 2

(2.1) A Sudanese woman has been freed from prison a month after being sentenced to die by hanging for refusing to renounce her Christian faith.

(2.2) "I am a Christian," Meriam Yehya Ibrahim told the judge at her sentencing hearing in May, "and I will remain a Christian."

(2.3) An appeals court in Sudan ruled that a lower court's judgment against the 27-year-old was faulty, her lawyer, Mohaned [sic?] Mustafa El-Nour, said Monday.

(2.4) He declined to elaborate.

(2.5) An international controversy erupted over Ibraham's conviction in May by a Sudanese court on charges of apostasy, or the renunciation of faith, and adultery.

(2.6) Ibrahim was eight months pregnant when was sentenced to suffer 100 lashes and then be hanged.

(2.7) "I'm so frustrated.

(2.8) I don't know what to do," her husband, Daniel Wani, told CNN in May.

(2.9) "I'm just praying."

(2.10) Wani uses a wheelchair and "totally depends on her for all details of his life," Ibrahim's lawyer said.

(2.11) Ibrahim was reunited with her husband after getting out of custody, her lawyer said Monday.

(2.12) Ibrahim gave birth to a girl in a prison last month, two weeks after she was sentenced.

(2.13) She was in the women's prison with her 20-month-old son, but Sudanese officials said the toddler was free to leave at any time, according to her lawyer.

(2.14) The criminal complaint filed by a brother, a Muslim, said her family was shocked to find out Ibrahim had married a Christian, U.S. citizen Daniel Wani, after she was missing for several years, according to her lawyer.

(2.15) A Muslim woman's marriage to a Christian man is not considered legal in Sudan, thus the adultery charge.

(Stories and sentences from Weston et al. 2015)