Story Visualization via NLP and Word Clouds

Emergent Research Forum (ERF)

Michel Mitri
James Madison University
mitrimx@jmu.edu

Abstract

This paper describes the use of two open source software libraries for extracting meaning from a story–like textual narrative and displaying it in an intuitive and interactive manner to users via a set of word cloud visualizations. The first library is Stanford's CoreNLP, a Java API that provides natural language processing services such as sentence recognition, tokenizing, parts-of-speech identification, dependency parsing, named entity recognition, and coreference resolution. The second is D3, a JavaScript API built on scalable vector graphics which provides powerful data visualization capabilities. The paper presents an application that uses CoreNLP to extract key story elements from a body of text (subjects, actions, objects, places, times, and other contextual features), and uses D3 to display these elements and their interactions in an integrated and interactive set of word clouds.

Keywords

Natural language processing, dependency parsing, coreference resolution, data visualization, word clouds, Stanford CoreNLP, D3.

Introduction

This paper describes a software application that helps users visualize a story by making use of natural language processing (NLP) software and data visualization. Here, the term "story" refers to a narrative that involves people, groups, organizations, or other entities (subjects) performing actions that can affect other people, organizations, or entities (objects). These events occur in certain places and at certain times. In the context of this paper, "understanding the meaning" of a story pertains to identifying these key elements of the story (subject, objects, actions, time, place, and other contexts), and moreover to represent the relationships between these elements for each event that takes place in the story. In other words, the software described in this paper attempts to visually and interactively answer this question: Who did what to whom, where and when did it happen, and what else was going on at the time?

Stories can be fictional or non-fictional. These run the gamut from novels, novellas, or short stories to newspaper or magazine articles. Note that this type of "text understanding" is very different from other kinds. Reading and understanding a story is not the same as, say, reading and understanding a technical manual or an anatomy textbook. I believe that NLP/NLU can be applied to many types of reading tasks, but for this paper we focus on story understanding, and especially the notion of subjects impacting objects at certain places and times.

There is a well-known saying: "A picture tells a thousand words." Visualization software can be used to draw a picture of this narrative via word clouds showing the interaction between subject, object, actions, places, times, and other contexts. This kind of visualization, leveraging on the NLP functionality discussed above, can help a user quickly capture the essence of a story.

Currently, word clouds are mostly used to depict term frequencies in documents. There is not much use of word clouds for depicting linguistic or semantic relationships, as this research attempts to do. The combination of NLP and word cloud visualizations helps to bridge a gap between the NLP and visualization research communities.

Natural language processing (NLP) is a branch of artificial intelligence, computer science, and computational linguistics. NLP systems perform many tasks necessary for making sense of text or speech recognition. Some of these are grammatically focused, such as parts-of-speech (POS) tagging and syntactic parsing. Others are based on recognizing co-occurrences of entities in a document (coreference resolution) or recognizing named entities. At a deeper level, NLP systems attempt to extract or infer the underlying meaning of text; this is often termed natural language understanding (NLU). The extension of NLP to include more abstract, higher-level text understanding (NLU), as this research attempts to do with story element extraction, also helps to fill a gap in the current state of the art.

NLP has advanced to the point that there are powerful open source APIs available, such as Apache's OpenNLP and Stanford's CoreNLP. These APIs include models for performing the above-mentioned tasks. Many of these models are generated through machine learning algorithms, trained on large quantities of text documents; others are more rule-based (Manning et al, 2014).

NLP/NLU systems, when combined with data visualization tools, can assist users to rapidly understand a body of text. For example, the output of NLP functions can be combined with word clouds to help users quickly capture the underlying meaning of a body of text. In addition to being a useful wat of visualizing text information, this technology potentially be could be used to assist non-English speakers or people with reading disabilities such as dyslexia to quickly grasp the essence of a story. Note that CoreNLP has annotators for multiple languages: Arabic, Chinese, French, German, and Spanish. (https://stanfordnlp.github.io/CoreNLP/)

The following sections describe an application that combines CoreNLP with D3 (Zhu 2013), a JavaScript API based on scalable vector graphics (Lane 2015). The application attempts to extract key elements of a textual story, and display these in word clouds.

Using CoreNLP to Identify Story Elements

Stanford's CoreNLP (Manning et al, 2014) is a Java API consisting of a suite of tools called *annotators* that can perform many NLP functions. The annotators provide several useful services, including the following:

- Breaking a text document into individual sentences
- Tokenizing a sentence (breaking it into individual "words")
- Identifying parts of speech (POS) within a sentence (nouns, verbs, adjectives, adverbs, etc.)
- Named entity recognition – recognizing names of people, places, organizations, dates/times, etc.
- Constituency parsing – constructing taxonomies of noun phrases and verb phrases of a sentence
- Dependency parsing – constructing the graph of dependency relationships between terms in a sentence
- Co-reference resolution – finding all expressions that refer to the same entity in a text

Using CoreNLP, one can build software to facilitate story understanding. The above annotators help to extract higher-level meaning from text. The application described in this paper attempts to identify basic elements of a story and display them in an intuitively appealing and interactive manner to users.

Consider the following text:

> *Before summer of 2016, almost nobody expected Donald J. Trump to be our next president. Most pundits expected Hillary Clinton to win. But over time, Trump appealed to a core base of die-hard supporters, and these supporters proved us wrong. During the Republican primaries, Trump beat Marco Rubio in Florida and was defeated by John Kasich in Ohio. Throughout the campaign, he repeatedly referred to Rubio as "Little Marco", and he castigated Ted Cruz as "Lyin' Ted". Contrary to conventional wisdom, Trump beat Kasich, Rubio, Cruz, and all the other presidential candidates at the Republican national convention in Cleveland, Ohio in July.*
>
> *Also in July, Hillary Clinton defeated Bernie Sanders for the democratic nomination. Ultimately, Trump beat Hillary in the general election on Nov. 8, 2017, although Clinton beat Trump in Virginia and in Maryland. Overall, the popular vote favored Clinton, but the electoral college chose Trump. Since becoming president in January, Trump selected several cabinet appointees, he fought with the press (who he calls the "dishonest media"), and he tweeted*

> *several times to the American people. In February, congress approved many of Trump's cabinet appointees.*
>
> *Also in February, Trump fired Mike Flynn because Flynn misled the vice president in January about his conversations with Russian officials in December. The American people have elected Donald Trump to be the next president of the United States. Now "the Donald" is leading us, and we are apprehensively watching to see where he takes us.*

After reading this, you have a sense of the story. This includes the actions that took place (verbs), the people or entities who performed these actions (subjects) and the people or entities who were affected (objects). You also have a sense of when and where these actions took place, as well as other contextual features. The following paragraphs and figures describe how we can use CoreNLP to extract each of these six story elements (subjects, actions, objects, places, times, and other contexts) for a story like this. Use of each annotator will be discussed and shown.

The first two CoreNLP annotators are *sentence splitting* and *tokenization*. Sentence splitting seems pretty straightforward because sentences generally end in periods, question marks, or exclamation points. But there are some tricks to consider; for example, periods at the end abbreviations (e.g. Mr. or Ms. or Dr.). Tokenization means splitting the sentence into individual words. Here, spaces are obvious delimiters, but also word contractions must be recognized (e.g. won't or I'll).

The next annotator is *parts-of-speech (POS) tagging*. CoreNLP's POS annotator identifies words as nouns, verbs, adjectives, etc. according to the Penn Treebank tag set (Marcus et al, 1993). POS results are important for identifying the actions (verbs). Also note that the subjects and objects of these actions are likely to be nouns. Also important is the *named entity recognition (NER)* annotator. NER is especially useful for story understanding, because it identifies words and phrases as people, locations, organizations, or dates and times, all of which are key story elements.

The most essential CoreNLP annotator for identifying the subjects and objects of actions is the *dependency parser*. The dependency parser breaks a sentence into dependency relations, which are binary predicates involving two words, a governor and a dependent (de Marneffe and Manning, 2008). Two key dependency relations of an active voice sentence are the nominal subject (nsubj) and the direct object (dobj). If the same verb is the governor of both nsubj and dobj relations of the same sentence, then the nsubj dependent is the subject of that verb and the dobj dependent is its object. Similarly, passive voice sentences will involve an agent relation and a passive nominal subject relation.

For example, consider the sentence "The dog bit the mailman." Also consider the sentence "The mailman was bit by the dog." Both are depicted in Figure 1.

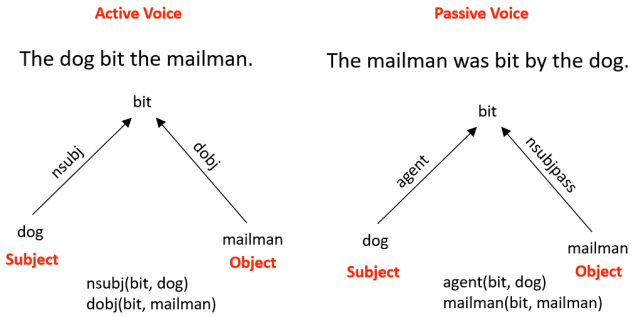

Figure 1. Subject and object dependencies from CoreNLP

CoreNLP's dependency parser recognizes over fifty different types of dependency relations. A sentence will typically have several dependency relations between words. Some are particularly useful in identifying the connection between subjects and objects of complex sentences, or for more completely depicting a subject or object. There include adjectival modifiers (amod), noun modifiers (nn), prepositional modifiers (prep), verb modifiers (vmod), and open clausal complements (xcomp). The full set of relations in a sentence forms a graph, and the process of connecting subjects to objects involves a recursive traversal of this graph (Feng et al, 2015)focusing on the most relevant dependencies. As shown in Figure 2, CoreNLP's

dependency annotator provides an XML representation of each sentence's dependency graph, which allows use of XPath and Java XML libraries for querying and searching the dependencies.

Another important CoreNLP annotator is *coreference resolution*. This associates words and phrases in the text that refer to the *same* entity. For example, in the above narrative, there are many sentences that refer to Trump. His full name first comes up in sentence 1, but then there are many other references to him. When applied to the above story, CoreNLP's coreference resolution annotator identified 15 mentions of Trump. Sometimes they are the same name, sometimes only first or last names, and sometimes the pronoun "he". Coreference resolution is far from perfect; current tests give it only 60-70% accuracy rates, and there is a heavy tradeoff between precision (avoiding false positives) and recall avoiding false negatives) (see, for example, https://nlp.stanford.edu/software/dcoref.shtml). Nevertheless coreference resolution is essential for producing a coherent understanding of a story when the same subject or object is referenced many times using different labels throughout the text.

Figure 2 shows a screen shot of a Java application that uses all these CoreNLP annotators to extract the meaning from the story. This figure shows the dependencies, tokens, parts of speech, and recognized named entities of a sentence in the story. The application also traces and displays, for each verb, the dependency paths between subject and object, as well as the coreference chains of the full story.

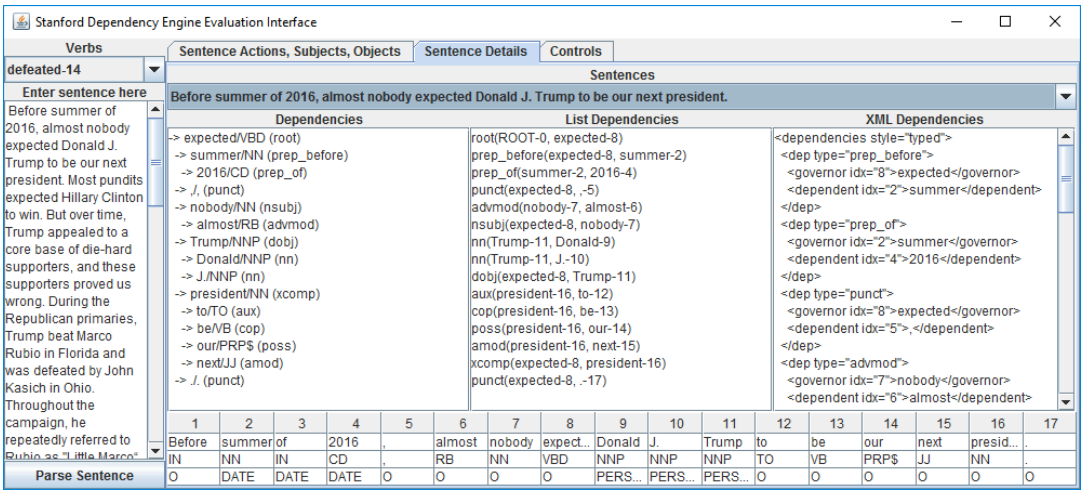

Figure 2. Application showing dependencies, parts-of-speech, and named entities from CoreNLP.

Interactive Word Cloud Visualization using D3

The previous section described a Java application that uses CoreNLP to extract the subject/verb/object/place/time/context associations of a textual story. This application is shown in Figure 2. After extracting these story elements, the application uses D3 to produce an HTML/JavaScript file that displays word clouds for visualizing each of these elements. This is shown in Figure 3.

Here you see six word clouds. Each word or phrase in a word cloud is displayed with its position within the sentence and sentence number of its first occurrence. Here Trump's first occurrence is in sentence 1, position 11. Within a cloud, font size indicates impact or prevalence. Above, Trump is bigger than other subjects because of all the actions (verbs) he does and all the affected objects.

When the user hovers the mouse over a word in one cloud, the visualization highlights its related words in the other clouds. Above, Trump is selected in the subjects cloud, and appears in black. All the actions done by Trump, their affected objects, and the relevant places, times, and other contexts are highlighted. Color is used to identify the specific relationships. For example, the dark green words refer to Trump's action in sentence #6. Specifically, he beat (verb), Rubio, Kasich, and other presidential candidates (objects), in Cleveland (place), in July (time), at the Republican national convention (other contexts).

Trump is also prominent as an object, as we see in Figure 4. Several things happened to Trump, by various subjects at various places and times, which is why his name is also the largest in the objects cloud.

Again, color coding associates the actions, subjects, and other contextual elements. Selecting a word in any word clouds displays associations for that word.

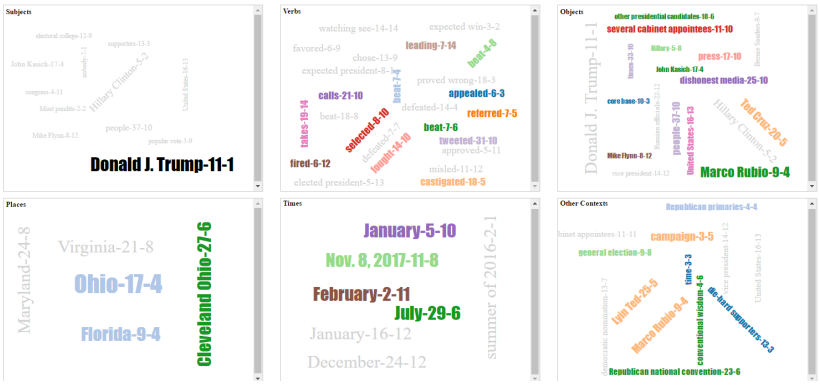

Figure 3. D3-generated word clouds for story visualization (subject selected)

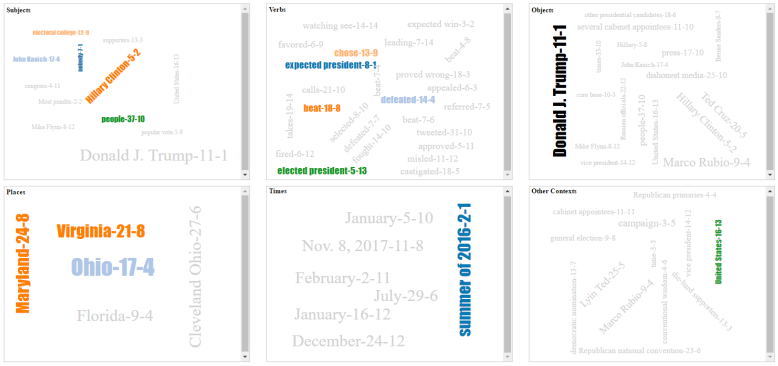

Figure 4. D3-generated word clouds for story visualization (object selected)

Conclusion

This paper presents an application that combines CoreNLP and D3 functionality to assist users in quickly understanding the basic elements of a story, employing the maxim "a picture tells a thousand words." The application uses CoreNLP to split the text into sentences, tokenize the sentences, identify parts-of-speech and named entities, and find coreferences. The application also traverses each sentence's dependency network generated by CoreNLP's syntactic parser. Using these tools, the program attempts to extract story's subjects, actions, and objects, as well as each action's contextual factors such as place, time, and prepositional elements. D3 is then used to display these in word clouds. This application demonstrates the promise of NLP and visualization technologies for facilitating natural language understanding (NLU), and decision support of unstructured data.

There remain many limitations and challenges. The current state of NLP (and especially NLU) is far less advanced and reliable than working with structured data. In addition, the application described here is limited to a narrow branch of text understanding; it applies only to understanding a story. The application is not well suited for extracting meaning from other types of text (e.g. ascertaining taxonomic hierarchies, understanding philosophical readings, etc.).

Nevertheless, as technologies advance, and developers become more familiar with the tools available, opportunities for building useful software for extracting and presenting meaning from text will continue to grow.

Acknowledgements

The author would like to thank James Madison University's College of Business for awarding a fall 2015 paid educational leave, which provided valuable time and resources for pursuing this project.

REFERENCES

de Marneffe, M-C and Manning, C.D. 2008. "Stanford Typed Dependencies Manual", http://nlp.stanford.edu/software/dependencies_manual.pdf.

Feng, Y., Deng, Q., Yu, D. 2015. "BLCUNLP: Corpus Pattern Analysis for Verbs Based on Dependency Chain" in *9th International Workshop of Semantic Evaluation; Proceedings of SemEval-2015*, June 4-5 2015, Denver, CO, pp.325-328.

Lane, D. 2015. "Scalable Vector Graphics", *Journal of Online Mathematics and its Applcations (7)*, article ID 1381.

Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.. 2014. "The Stanford CoreNLP Natural Language Processing Toolkit" in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55-60.

Marcus, M.P.., Marcinkiewicz, M.A., Santorini, B. 1993. "Building a Large Annotated Corpus of English: The Penn Treebank" Computational Linguistics (19:2), pp. 313-330.

Zhu, N.Q. 2013. *Data Visualization with D3.js Cookbook*, Packt Publishing, Ltd.