NLP Summary

Jesse Shellabarger
John Kirschenheiter

Procedure

Our software begins by immediately parsing the Lincoln document. The curated document is stored with the software, so the process does not to be repeated every time the software runs. We use the Stanford Natural Language Processor (https://nlp.stanford.edu/software/) to parse every sentence within the article. The natural language processor is able to split the document into individual sentences. We loop over each of these sentences and find dependencies between words in the sentence with the parser. Using these dependencies, we are able to identify the subject, verb, and direct object of each sentence. We use the Stanford processor to perform stemming and lemmatization on each word, giving us the most generic and consistent form of the words. For example, we would store the verb "runs" in its infinitive "run". We then store this sentence structure in a data structure (SentenceParse.java) of our own making, which only provides a wrapper for the three sentence components. We store a list of SentenceParse objects that we can reference each time the user enters a query. This process takes approximately thirty seconds on our school laptops.

After the document has been parsed, we allow the user to begin entering queries. We use the same procedure, using the Stanford parser, to create SentenceParse objects representing the query. We then compare the SentenceParse created from the query to every SentenceParse created from the Lincoln article. If any of them match, we know that the query and a sentence in the query and a sentence in the article have similar structure and content. Thus, the query is likely true. If none of the SentenceParses from the document match, then the query is likely false.

Curation

Our curation process was very similar to the one we used in the Information Retrieval assignment. First, we used an online tool (https://www.phpjunkyard.com/tools/html-to-text.php) to convert the html from the wikipedia page into plain text. We then removed some of the text pertaining only to Wikipedia and not to the article. This included the table of contents, the reference list, and the summary box. We also removed the title of the page because it contained no relevant factual information. We also removed any of the lines that seemed to have excessive

white space. These lines were a result of the online html converter we used, and mostly referenced unuseful data.

Several changes were made from our Information Retrieval assignment curation process. This time, we left sentence ending punctuation in the document. This helped to Stanford natural language processing library to identify sentences. We did find it interesting, however, that the parser was able to separate the sentences in the document without punctuation. It's accuracy was not as high, but this is impressive nonetheless. We did remove other punctuation in the sentence such as commas, parentheses, and quotes. We found that this additional punctuation did not improve the parser's accuracy by much but did make it harder for us to parse the results.

Summary of Results

We had very mixed results when it came to producing good results. While this method works well for simple sentences with the sentence structure we expected, it fell short whenever it had to describe an input query with a more complicated structure. Additionally, the Stanford parser was good, but not perfect. This was less of concern when processing the Lincoln document, but if an error occurred while searching the input query, it occasionally lead to unexpected errors.

Another source of unexpected errors is the short search space. The Lincoln document isn't short for an wikipedia article, but it is our only source of information. This means that there are certain sentences that we expected to be there, but just weren't when we ran a search. A good example of this is the sentence "Lincoln ran for president," which we expected to produce results. This sentence doesn't actually appear in the document, so the search correctly came back negative. This problem was exaggerated by the sentence structure of the article, which are often complicated and hard to parse.

Altogether, our software works decently well for direct queries with a very clear subject and verb. That being said, the software occasionally fails even with these types inputs. With more complicated inputs, the function has almost no hope of successfully returning a positive answer.

Our software is also only able to evaluate queries that are one sentence in length. This is an intentional limitation, rather than a technical one. We found that allowing a user to enter multiple sentences at a time only cluttered our output. Instead, we simply force the user to enter their queries one at a time. We are able to parse the entire Lincoln document (a lengthy document) in under a minute.