NLP Final Project Report

Tobin Yehle and Dasha Pruss

December 10, 2015

1 Components and Contributions

All of the NLP code is in `Sherlock.java`. We have a `main` method inside of `Driver.java`, which does all the IO and data wrangling. We made some data structures for storing stories, which are in `Story.java`. Most of this code was auto-generated (`equals`, `toString`, etc.).

Our `Sherlock.java` file contains a single public function, `processStory`, which calls the numerous private methods that we wrote to first find the best sentence in the document, and then extract key phrases from that sentence.

We pair programmed our entire project and each of us spent roughly equal time programming. The only file of real substance was `Sherlock.java`, and it underwent many changes. If you would like to see the specifics of who edited what, feel free to browse the commit history of our project in our bit bucket repository. Generally the name on the commit was the person typing, while the other team member dished out advice. To see who last edited each line of a file you can take a look at blame.

2 External Resources

We used Stanford's CoreNLP libraries for tokenization, sentence splitting, part-of-speech tagging, NER, parsing, and coreference resolution. We used their models that were trained with news article corpora, such as CoNLL.

3 Regrets and Successes

We regret spending so much of our time trying to get co-reference resolution to work because it ended up not helping us at all — in fact, it made our f-score worse. Instead of spending our time on coreference, we wish we had used WordNet to find semantic classes our of noun phrases to improve our phrase extraction procedure.

The rules that we implemented, both from the Quarc paper as well as our own rules, were highly effective at identifying the correct sentence. We were also happy with the way we extracted phrases from the sentences — it was a very simple approach, but surprisingly effective.