Proposal Assignment 2

Group Members (Group 50): Clifton Roozendal, Floris ten Lohuis

For assignment 2 we would like to further build on the program we have set up for assignment 1. Our motivation behind this is several fold:

1. We have set up a solid base on which we can further build
   1. The program has been built in a modular way so that we can fine-tune individual parts of the program
   2. We have parameterized the various modules to easily be able to adjust settings and implementations
2. We have already pinpointed several items which we thought could be (further) improved in our report for assignment 1 (see appendix for the items which we have already mentioned). Actually tackling these for assignment 2 gives us the perfect opportunity to make the program more robust and improve its performance. It will also allow us to further master this part of the course which we all found very intriguing

Our roadmap will be to implement several additional steps and implementations in the (pre-)processing (cleaning, searching, etc.) and get the trident filtering of the code working. We will then build a grid-search to test and further tune the model. It will in that sense be an iterative improvement process, with the ultimate goal of delivering a more robust, complete and better performing model.

As we will be working in a smaller group, we expect to be working very closely together throughout the assignment. For efficiency’s sake we will of course split up and pick up individual modules in the process but will (continue to) reconvene frequently to discuss and spar about what we have built.

# Appendix

The italicised section below is a direct excerpt from our report from assignment 1. This has been added to give additional context about the sections which we would like to (further) improve.

# *Next Steps / Future Releases*

*The current version of the program performs quite efficiently, with scalability in mind, and can process the given input through to returning the refined Wikilink hits. It has also been set up in a modular way so that specific parts of the pipeline can be further tuned without affecting earlier or later parts of the calculation. Some of the things which could be further fine-tuned are:*

* *Cleaning*
  + *Try other (custom) HTML parsers*
* *Extract*
  + *Extend the current RegEx rules for further cleaning*
  + *Use more parts of the Spacy pipeline*
  + *Try different extraction models*
    - *Both different models within Spacy as well as entirely different NLP processors*
  + *Run similarity matching in parallel to improve the speed*
  + *Fine-tune similarity cut-off*
* *Search*
  + *Fix asynchronous searches to optimize speed*
  + *Improve ElasticSearch query speed*
  + *Finetune ElasticSearch queries (for example, look at whether combining multiple words in {bool: {should : [...]}} queries improves results)*
  + *Finetune number of hits being returned*
* *Decision*
  + *Get the class working*
  + *Tune the number of records which should be kept (or base it on different criteria)*