Adam Horvath-Smith

CS 7800 Project 1 Review

The design of the project was constrained to the skeleton code given from the start. I used the classes and function names given to me to ensure that there would be consistency with test cases used in grading and review. I did add a few helper methods when implementing the skeleton functions, but these never need to be called directly; these are only ever called from within the functions defined in the original skeleton code. Using the skeleton code layout left me with a rather modular and re-usable design. All parts of the project are neatly separated into separate Python files. Functions that were needed in several parts of the project were placed in the util.py file for easy access.

The code was implemented according to the algorithms presented in class notes and in the course textbook. I included a link to the textbook's definition of the algorithm in my code when I used its implementation. After finding the algorithms, these were implemented as closely as possible to the textbook definitions. There are a few changes from the original implementations, however. For example, the SPIMI algorithm I implemented omitted saving the index in blocks when the index was generated. Using the given algorithms to build the search engine made the implementation rather straightforward.

When it came time to test my implementation of the search engine, I wrote several separate test cases for each part of the project. Generally, I would test each individual function once its implementation was complete. For the most part, these test cases are still in the "test()" functions in both index.py and query.py, though I did do some early testing directly from the command line. Each of these test cases should still pass; simply call test() from either index.py or query.py to see the results. My tests included most of the points mentioned in the project requirements, like checking whether stopwords are removed or ensuring TF-IDF was computed correctly. In addition, I created several other test cases on my own. For example, I created several sample Boolean queries to ensure that this produced the same results as I computed by hand. For vector queries, I simply ran through several of the given queries and made sure some of the results matched those in qrels.text.