Cormick Hnilicka

Garritt Moede

Evan Kivolowitz

How did you develop the final blocker? What blocker did you start with? What problems did you see? Then how did you revise it to come up with the next blocker? In short, explain the *development process*, from the first blocker all the way to the final blocker (that you submit in the Jupyter file).

Our blocking stage was very crucial in our development process because out total number of possible combination was just about 6.6 billion (see below). We wanted to make sure we could prune as many unlikely combinations as possible before proceeding to the matching stage. In order to do this, we tried a couple of different blocking techniques. Blocking on the attribute "title" makes the most intuitive sense because if two tuples have radically different titles, they are very unlikely to match. We first blocked using the Jaccard Score with a threshold of 0.5. Our tokenization technique was to separate at word boundaries (whitespace). Although this was successful in reducing our lists to a candidate set of 385,009, we aimed to reduce it even more. After changing the threshold to 0.7 with the same parameters as before, we were able to reduce it to a candidate size of 51,322. This is significantly reduced from our original problem size, however, we did wish to see how the blocking performed with a q-gram tokenization. We again performed blocking on the Jaccard Score with a threshold of 0.5 and 0.7 again and the most successful was a candidate set of 50,109. This was achieved with a 3-gram tokenization and a threshold of 0.7. We will proceed with this candidate set for the matching stage.

If you use Magellan, then did you use the debugger? If so, where in the process? And what did you find? Was it useful, in what way? If you do not use Magellan, you can skip this question.

We attempted to use the debugger, however, we continually got an error saying DataFrame information is not presen t in the catalog. We were unable to resolve the issue.

How much time did it take for you to do the whole blocking process?

The entire blocking process took a couple days to complete. Due to our massive possible solution size of 6.6 billion, some of our blocking commands took several minutes to complete. Taking that into account, testing multiple blocking techniques compounded upon one another making the entire process take much longer than we anticipated.

Report the size of table A, the size of table B, the total number of tuple pairs in the Catersian product of A and B, and the total number of tuple pairs in the table C.

Our table A was taken from the music data dump and had a total number of 347,258 tuples. Our table B was comprised of tuples that we scraped from the web and contained 19,153 tuples. This brings our Cartesian product of the two to a massive 6,651,032,474 total number of possible combinations. Having this large of a possible solution spaces requires us to proceed with blocking before the matching stage. Matching with a total possible number of combinations of 6.6 billion would take far too long. After performing our blocking on the the song title, we were able to reduce the candidate set down to 50,109. This is a massive reduction from the original possible solution size.

Did you have to do any cleaning or additional information extraction on tables A and B?

During the blocking process, we did run into issues that forced us to perform further cleaning. For example, there was a handful of tuples in our musicbrainz_final.csv that ended up having 5 attributes instead of our intended 4. This was caused by some artists and song titles also containing a comma in them that slipped by our initial cleaning. We

simply removed the 390 tuples from the original 19,544 that fell under this criteria and the rest of the process was fairly straightforward.

Did you run into any issues using Magellan (such as scalability?). Provide feedback on Magellan. Is there anything you want to see in Magellan (and is not there)? If you do not use Magellan, you can skip this question.

For the most part Magellan was straightforward. However, we did have a great deal of issues when attempting to install and/or running the py_entitymatching python package. We continually got an error saying "Catalog was not preset in py_entitymatching." We eventually resolved this issue by running the setup.py file in Python3 instead of the default python command. We also had a brief issue with our DataFrames being constructed properly when reading in the CSV files. However, this was a mistake on our part as we had spaces (" ") present in the attribute names. Once this was corrected, py_entitymatching seemed to function properly.

Any other feedback is appreciated

Magellen was very helpful during the blocking phase of the project because it allowed us to begin blocking immediately. The environment set up by Magellen gave us the opportunity to try many different blocking techniques without having to spend much time getting them up and running. The only thing that we would have appreciated a bit more is if the tutorial provided for using py_entitymatching in the Jupyter notebook provided more details and insight as to what is happening at each step of the way.