Natural Language Processing Project
Midterm Report

Group 1c

April 2020

In this report, we compare Sent2Vec Search, Hybrid Search, and Elastic Search. The queries are titles, and the documents are abstracts.

1 Data Preparation

For this task, we used the reduced dataset of 10,000 instances. Looking at the data, we found that a lot of the instances had missing entries. Our team decided that such instances were faulty, and excluded them from our search. As a result, we were left with 8573.
Then the data was lower cased, and tokenized using Snowball Stemmer.

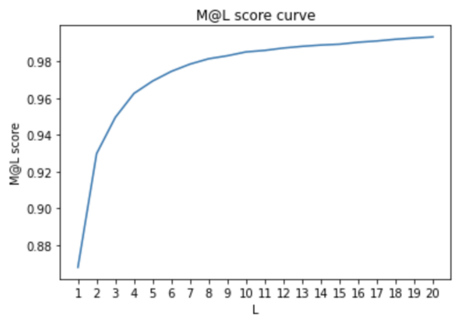
2 Analysis

2.1 Sent2Vec Search

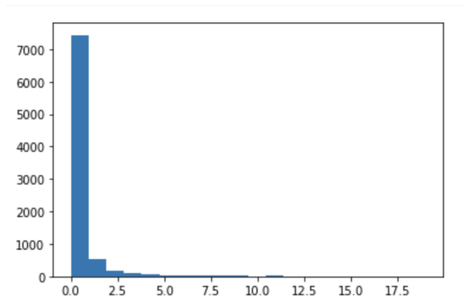
The first engine we used was sent2vec embedding. Using the sent2vec model, we embedded the titles and abstracts, and normalized them. Then similarity was computed using cosine similarity. The predictions took 28 seconds (including the computing time for the normalized embeddings of document list). The M@L score curve can be seen in figure 1a, with M@20 score being 0.9934 and M@1 being 0.8678.
The MRR score for the sent2vec engine was 0.9128

2.2 Hybrid Search

The second engine we used was hybrid search. Hybrid search combines keyword based boolean filtering with sent2vec based embedding. When evaluating the performance, our team decided that if boolean filtering returned no valid candidates (meaning every abstract was filtered out), we would mark it as a failure. We used 2 boolean logics for keyword based filtering; ‘and’, and ‘or’.



(a) M@L score

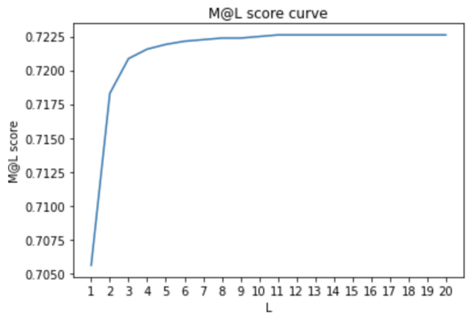


(b) M distribution

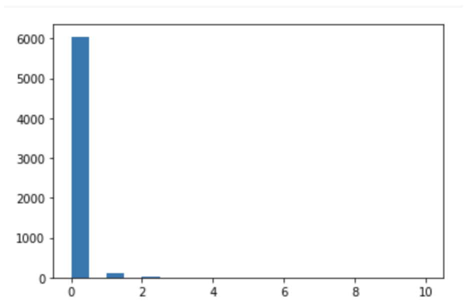
Figure 1: Sent2Vec Search performance

2.2.1 ’AND’ logic

The ‘and’ logic took 39 seconds (30 seconds for computing the inverted index, + 9 seconds for the search itself). It filtered out the whole abstract numerous times, so the search itself was quite fast. But its downside is that it gave a disappointing search result, shown in figure 2a with M@20 score being 0.7226 and M@1 score being 0.7056.



(a) M@L score



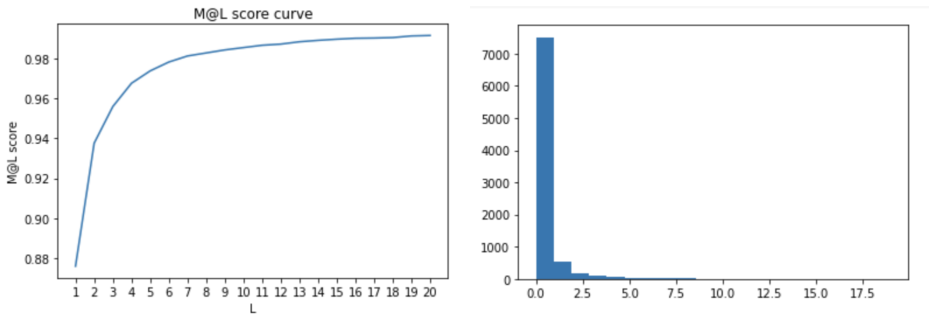
(b) M distribution

Figure 2: Hybrid Search performance (AND logic)

Its MRR score was 0.7131.

2.2.2 ’OR’ logic

The ‘or’ logic had more candidates to search compared to ’and’ logic, and took 77 seconds (30 seconds for computing the inverted index, + 47 seconds for the search itself). It gave promising search results, compared to ’and’ logic. The curve is given in figure 3a. M@20 score was 0.9915, with M@1 0.8760.
The MRR score for the Hybrid Search with ’or’ logic engine was 0.9191

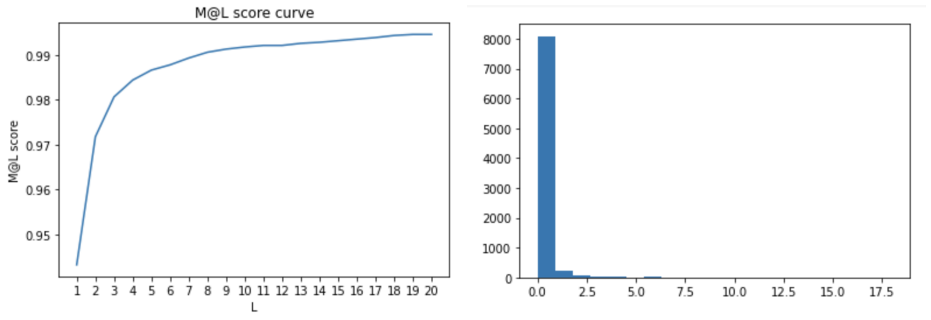


(a) M@L score **(b)** M distribution

Figure 3: Hybrid Search performance (OR logic)

2.3 Elastic Search

The last method our team used was Elastic Search. The search took 125 seconds. The result was the best out of the three methods. The curve can be seen in figure 4a. M@20 score for this search was 0.9945, with M@1 score of 0.9433.



(a) M@L score **(b)** M distribution

Figure 4: Elastic Search performance

The MRR score for Elastic Search was 0.9628.

3 Conclusion

Comparing the three search methods, the results can be interpreted as follows. Hybrid search with 'and' logic filtered out too many documents during boolean search, so it was the least accurate. Hybrid search with 'or' logic relaxed the boolean filter and gave a much more accurate result. Comparing hybrid-or search with sent2vec search, the latter gave a better M@20 score, but the former returned a higher M@1 and MRR score. We can interpret this as a result of boolean filtering. Boolean filtering reduced the number of documents to be searched per query, so if the target document was in the reduced set of documents, it is more likely that the target document will be the closest, hence higher M@1 score. Elastic search has much more functions built into it, so it takes a bit more time than sent2vec or hybrid search, and returns the best result. However it is worth noting that elastic search is considered a very fast search engine when dealing with a huge amount of data.