Assignment 2

Anonymous

***Abstract*—I present a selection of ranking methods, which include BM25, BM25 + query extension and BM25 + word embeddings. Those ranking methods then will be run on the Robust04 data set which includes more then 500'000 Documents and 250 Queries. So I'm going to show and compare the resulting performance for all 3 named methods and discuss about those results.**

***Index Terms*—BM25, Word2vec, query extension, word embeddings**

I. Introduction

The problem of ad-hoc retrieval is to be able to find relevant documents according to a query. This query normally consists of a few words from which we can find documents which consists of those words or similar words depending on the approach. In my example here we use the topics titles as the queries for the documents.

II. Method

BM25

I calculated the ranking to BM25 according to the following formula. In which we calculate with relevance information (r_i) and the frequencies (f_i), the parameters b,k_1,k_2 need to be selected accordingly. The variables dl is the document length and $avdl$ is the average document length.

$$score(D,Q) = \tag{1}$$

$$\sum_{i \in Q} IDF(q_i) \cdot \frac{(k_1+1)f_i}{K+f_i} \cdot \frac{(k_2+1)qf_i}{k_2+qf_i} \tag{2}$$

$$K = k_1((1-b)+b \cdot \frac{dl}{avdl}) \tag{3}$$

With this ranking calculation we will get a ranking for each query for all the documents.

BM25 Query extension

With this method I will use the same BM25 method as described above. The only thing that changes is the query itself, so we will add additional words which can be helpful for the improvement of finding relevant documents.

I use the centroid approach which takes the whole query into consideration for finding similar words.

$$S_{Cent}(t;q) = exp(cos(t, qCent)). \tag{4}$$

Additionaly I use a fusion based method approach by finding a similar words for each query term (the amount can be decided on).

$$S_{CombMAX}(t;q) = max_{q_i \in q} p(t|q_i). \tag{5}$$

A. BM25 with word embeddings

By using embeddings we use the following method, which looks very similar to BM25 with the difference that we multiplicate by the semantic similarity.

$$f_{sts}(s_l, s_s) = \tag{6}$$

$$\sum_{w \in s_t} IDF(w) \cdot \frac{sem(w,s_s) \cdot (k_1+1)}{sem(w,s_s) + k_1 \cdot (1-b+b \cdot \frac{|s_s|}{avgsl})} \tag{7}$$

The semantic similarity is defined as such:

$$sem(w,s) = max_{w' \in s}(w,w') \tag{8}$$

So in this case I compare the two terms by their cosine similarity which tells me the semantic similarity, which I then use in the above named math function.

III. Experiments

Before any of the following described experiments can be used on the dataset there needs to be some preprocessing happening on the data. By merging all the split up documents into a single file on which then the experiments can be run.

A. Bm25 Ranking for all Document

As a first experiment I ranked all documents by BM25 and saved the results of that on which then can `trec_eval` can be run.

B. BM25 Query extension

For this method I used the `Google News` Word2vec embeddings to find similar words acccording to which I extended the query. With those queries I then run the querying process with BM25 again to get the results of the run with expanded queries and evaluate it with `trec_eval`.

C. BM25 Embeddings

Use the function described from methods and also run the normal queries (not expanded) and expanded on with the new ranking method. From that we get 2 runs which we then can compare to other runs. But not implemented yet.

D. Cross validation

Sadly at this point the cross validation hasn't been implemented yet, so no tuning for BM25 yet.

IV. Results

When running all the ranking and then by evaluating with `eval_trac` I ended up on the following MAP (Mean Average Precision) for cut offs and all of the queries. For BM25+QE I included the centroid (C) and fusion (F) mthod I implemented. Word embeddings is work in progress and not finished yet, so the result is missing.

MAP	**BM25**	**BM25+QE**	**BM25+WE**
All Topics	0.0673	C: 0.0640, F: 0.0526	to do
Cut off 5	0.0961	C: 0.0935, F: 0.0774	to do
Cut off 10	0.1131	C: 0.1094, F: 0.0935	to do
Cut off 20	0.1252	C: 0.1210, F: 0.1040	to do

Additionally a result for folds would be helpful as well but there is not a implementation for cross validation yet.

Discussion

According to the results I get by using Query extensions can give better results but strangely the centroid approach works better then the fusion approach (which is also a very simple approach) seems to be less great to use. But as long as Word embeddings is missing for the results an over all conclusion can't be full done.

References

[1] Saar Kuzi, Anna Shtok, Oren Kurland, Query Expansion Using Word Embeddings

[2] Tom Kenter, Maarten de Rijke, Short Text Similarity with Word Embeddings

[3] Okapi BM25, https://en.wikipedia.org/wiki/Okapi_BM25