Algorithmic Details

Probabilistic Reranking

We first extract all the command line arguments using sys.argv[x] where x= 1,2,3 or 4. Then we open the top100 file and store the top 100 docs for the all the queries. Then we open the doc file and find the body/text corresponding to the documents given in top100 file. Them we open the query file to extract queries and corresponding query id. We use nltk tokenizer for tokenizing, nltk stopwords for stopword removal and krovetzstemmer for stemming as done in the original model. We store the queries and required documents in a dictionary with query id and docid as the respectively keys. In a separate dictionary we also keep track of docids corresponding to each query.

Now for each query we do the following. Convert the body/text of the 100 docs corresponding to that query into an inverted indexing structure as done in the previous assignment. In the indexing dictionary, we store all words as keys and a dictionary docids as the value. In the dictionary of docids corresponding to each word/term, we store their term frequency as the value. Now we iterate over all the words in the index to find the top terms to be added for query expansion. We do this by scoring each term using

VRi*wi   where VRi= term frequency of term in the relevant docs, wi=RSJ weight

We keep wi= round(math.log((N+2*df)/df),4)   N=Total number of docs, df=total document frequency

This is found by using pi= 1/3+(2/3)*df/N (Empirical formula determined by Grieff (1998)
And ui=df/N

Here we're considering the 100 docs as relevant and remaining as non-relevant. We can't iterate over whole 3.2 million doc collection to find df due to time bounds, so we use upscaling heuristic to approximate its value as df=VRi/100*N

Hence using this scoring, we find the top m terms where m is the expansion limit.

Now we find the bm25 scores of each query for all the docs by adding scores for their every term using the formula-
B25 score=wi*(tf*(1+k1))/(k1*(1-b+b*(dl/dlavg))+tf)
Here, wi=RSJ wt., tf=term frequency in that document, dl=document length, dlavg=average document length (averaged across top 100)
After finetuning, we found that k1=1.7, b=0.78

Then for we expand query term one by one, modify the scores, find the most relevant document and store it. We repeat this expansion_limit number of times.

Then finally we write out the results in m separate files corresponding to each expansion.

File name of results- result_pbx.txt   where x is the # terms added

Unigram

Initial steps of pre-processing remain the same. After creating the indexing, we do the following for every query.(Note that separate indexes of 100 docs is created for every query) –

We use the following formula for Bayesian smoothing-

P=log(mu*pc/(mu+dl))

Where mu is a constant who value generally kept as some multiple of average document length,

Here mu=3*dlavg

Pc is the probability score corresponding to the background language model which here is built on top of the top100 docs as collection

Pc=(# times that term appears in the collection)/(length of collection)

We add the scores for all terms in the query and find the doc with the max score. This is doc which is finally reported. We do this for all queries.

We finally write the results in result_uni.txt in trec eval format.

Bigram

Initial steps of pre-processing remain the same. However, in the indexing we also include bigrams along with unigrams/single terms. After creating the indexing, we do the following for every query (Note that separate indexes of 100 docs is created for every query) –

We use the following formula for scoring the docs:

$$\text{Bigram: } P(q) = P(w_1) \cdot P(w_2|w_1) \cdot P(w_3|\, w_2) \ldots$$

Here P(W1) is same as that used in the unigram model and for next terms, we use:

$$P(w_i|w_{i-1}, D) = (1-\lambda_1)\left[(1-\lambda_2)\frac{f_{w_i,w_{i-1},D}}{f_{w_{i-1},D}} + \lambda_2\left(\frac{f_{w_i,D}}{|D|}\right)\right] + \lambda_1\left[(1-\lambda_3)\left(\frac{cf_{w_i,w_{i-1}}}{cf_{w_{i-1}}}\right) + \lambda_3\frac{cf_{w_i}}{|C|}\right]$$

Here since we're using Bayesian smoothing, we put lamda=mu/(dl+mu)

Where mu=3*dlavg, same as in the unigram model.

We use same values for all lambda1, 2 and 3.

Here fwi,wi-1 =frequency of bigram term in the document and

Cfwi,wi-1=frequency of bigram term in the collection of 100 docs. Rest notations are similar to what used in the unigram model.

Since we're taking log therefore product changes to summation. We report the document with highest score for each query.

We finally write the results in result_bi.txt in trec eval format.

RESULTS

Probabilistic reranking

Number of terms added	Ndcg	MRR
1	0.1311	0.1311
2	0.1248	0.1248
3	0.1107	0.1107
4	0.0986	0.0986
5	0.0818	0.0818
6	0.0730	0.0730
7	0.0637	0.0637
8	0.0578	0.0578
9	0.0510	0.0510
10	0.0462	0.0462

Unigram

Ndcg=0.0701   MRR= 0.0701

Bigram

Ndcg= 0.0472 MRR= 0.0472

STATISTICAL TEST RESULTS

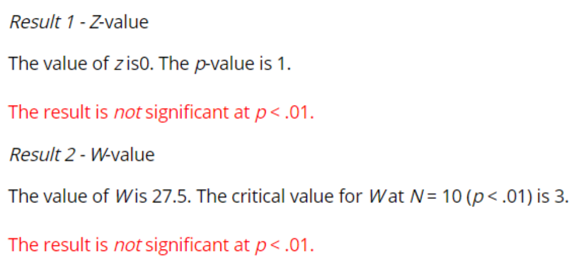
150 queries were used to calculate the given measures.

Significance Level=0.01, Two tailed Hypothesis

Wilcoxon Signed Rank Test

Site used: https://www.socscistatistics.com/tests/signedranks/default2.aspx

1)Models- Unigram and probabilistic query expansion(bm25)



Result 1 - *Z*-value

The value of *z* is0. The *p*-value is 1.

The result is *not* significant at *p* < .01.

Result 2 - *W*-value

The value of *W* is 27.5. The critical value for *W* at *N* = 10 (*p* < .01) is 3.

The result is *not* significant at *p* < .01.

2)Models- Bigram and Probabilistic query expansion(bm25)

Result 1 - Z-value

The value of *z* is-0.8745. The *p*-value is .3843.

The result is *not* significant at *p* < .01.

Result 2 - W-value

The value of *W* is 130. The critical value for *W* at *N* = 25 (*p* < .01) is 68.

The result is *not* significant at *p* < .01.

3)Models- Bigram and Unigram

Result 1 - Z-value

The value of *z* is-0.9124. The *p*-value is .36282.

The result is *not* significant at *p* < .01.

Result 2 - W-value

The value of *W* is 108. The critical value for *W* at *N* = 23 (*p* < .01) is 54.

The result is *not* significant at *p* < .01.

Paired Sample T Test

Site Used: https://www.socscistatistics.com/tests/ttestdependent/default2.aspx

1)Models- Unigram and probabilistic query expansion (bm25)

The value of *t* is 0. The value of *p* is 1. The result is *not* significant at p < .01.

2)Models- Bigram and Probabilistic query expansion(bm25)

The value of *t* is -1. The value of *p* is .31893. The result is *not* significant at p < .01.

3)Models- Bigram and Unigram

The value of *t* is -1.042876. The value of *p* is .29869. The result is *not* significant at p < .01.