Bonus submission for HW4

In this homework, we implemented both query expansion and pseudo relevance feedback.

We supported three different preprocessing modes which were used for query expansion

1. Stemming (no query expansion)
2. Lemmatization + Query Expansion
3. Lemmatization + POS tagging + Query Expansion

# Query Expansion

## Stemming

In Stemming, nltk.PorterStemmer is used and the following results were obtained.

>>>

Searching for the query "queries/q1.txt" ...

Execution time: 0.2s

6807771 : Rank 4469

4001247 : Rank 248

3992148 : Rank 3791

Searching for the query "queries/q2.txt" ...

Execution time: 0.2s

2211154 : Rank 90

2748529 : Rank 1

Searching for the query "queries/q3.txt" ...

Execution time: 0.3s

4273155 : Rank 18

3243674 : Rank 2

2702938 : Rank 16

## Lemmatization + Query Expansion

This preprocessing made use of WordNet which is a large lexical database of English, including nouns, verbs and adjectives. We support query expansion by making use of nltk.Lemmatizer which returns the dictionary form of a word, with respect to WordNet.

The following results were obtained

>>>

Searching for the query "queries/q1.txt" ...

Execution time: 1.6s

6807771 : Rank 2785

4001247 : Rank 62

3992148 : Rank 543

Searching for the query "queries/q2.txt" ...

Execution time: 1.6s

2211154 : Rank 75

2748529 : Rank 24

Searching for the query "queries/q3.txt" ...

Execution time: 1.7s

4273155 : Rank 5

3243674 : Rank 3

2702938 : Rank 9

Query expansion was performed by retrieving the synonyms of the query terms using WordNet.NOUN synsets interface which represents groups of synonymous words that express the same concept.

The usage of query expansion clearly improved performance on test case 1 which reflects the anomalous state of knowledge (ASK) problem or vocabulary mismatch wherein the queries use terminologies that do not fit the documents’ expression of the same semantics.

## Lemmatization + POS Tagging + Query Expansion

This preprocessing technique also uses the Part-of-Speech (POS) tagging which tags words in a corpus corresponding to a particular part of speech, based on its definition and context. This includes classifying words as nouns, verbs and so on. This thus technically is supposed to add more context to our query expansion as we now use the WordNet.synset with the stated POS for each query term instead of just NOUN.

The results are as follows:

>>>

Execution time: 1.6s

6807771 : Rank 4200

4001247 : Rank 88

3992148 : Rank 1373

Searching for the query "queries/q2.txt" ...

Execution time: 1.7s

2211154 : Rank 108

2748529 : Rank 32

Searching for the query "queries/q3.txt" ...

Execution time: 1.6s

4273155 : Rank 7

3243674 : Rank 3

2702938 : Rank 8

However, we observe that the performance on test case 1 now degrades. A possible reason is that we now add more query terms than necessary and flood the list of candidate documents, thus resulting in the relevant documents being pushed down.

# Pseudo Relevance Feedback

Pseudo relevance feedback was also implemented. We first obtained our list of candidate documents, assumed that the top n documents were relevant, and used the Rocchio formula to adjust the new query weights.

Although relevance feedback did marginally improve the results, the results were too dependent on the parameters (alpha, beta, gamma and num of docs) which we were unable to tune accordingly due to the lack of a sufficiently large training and validation dataset.