IR - Term Project

Group Number: 22

Group Members:
1.Arnab Kumar Mallick - 18CH10011
2. Arghyadeep Bandyopadhyay - 18EE10012
3. Subham Karmakar - 18EE10067
4. Sankalp Srivastava - 18EE10069

Task B1 (Relevance Feedback)

Performed by:
Arnab Kumar Mallick (Roll No: 18CH10011)
Arghyadeep Bandyopadhyay (Roll No: 18EE10012)
Subham Karmakar (Roll No: 18EE10067)
Sankalp Srivastava (Roll No: 18EE10069)

The intermediate files generated while running the commands mentioned below for Task B1 would be PB_22_ranked_list_RF_<K>, PB_22_ranked_list_PsRF_<K>, PB_22_rocchio_RF_evaluated_<K> and PB_22_rocchio_PsRF_evaluated_<K> where K = 0, 1, 2.
These correspond to the following:

- K = 0 is [$\alpha = 1$, $\beta = 1$ and $\gamma = 0.5$]
- K = 1 is [$\alpha = 0.5$, $\beta = 0.5$ and $\gamma = 0.5$]
- K = 2 is [$\alpha = 1$, $\beta = 0.5$ and $\gamma = 0$]

Steps used:

1. The paths and file names of all the documents are extracted from the ‘en_BDNews24’ folder.

2. The set of relevant documents for Pseudo Relevance Feedback (PsRF) and the sets of relevant and non-relevant documents for Relevance Feedback (RF) corresponding to each query are stored.

3. The inverted index file is read and the document frequencies (df) along with the vocabulary are stored.

4. For each document, the document text is stored as a list of terms using the inverted index.

5. The text file containing the queries is read and each query is then stored as a list of the terms contained in that query.

6. The normalised |V|-dimensional TF-IDF vectors are obtained for each query with the given weighting scheme (lnc.ltc).

7. The centroids of the normalised vectors of the documents in the relevant and non-relevant document sets are obtained corresponding to each query for both the RF and PsRF schemes (for PsRF, the set of non-relevant documents is null).

8. For each document, at first the |V|-dimensional TF-IDF vector is obtained with the given weighting scheme(lnc.ltc). Then, for each query vector, the dot products between the original query vector and the document vector, between the centroid of relevant documents for PsRF scheme and the document vector, between the centroid of relevant documents for RF scheme and the document vector, between the centroid of non-relevant documents for RF scheme and the document vector, are computed. The dot product of the document vector with the modified query vector for both RF and PsRF schemes are thus computed by adding the individual dot products with the corresponding weights (alpha/beta/gamma) in accordance with the equation in Rocchio's algorithm. This is done because if we compute the modified query vectors first and then compute the dot product, we need to compute the dot products for as many times as the number of values of alpha, beta and gamma. This is expensive as the modified query vectors can be quite big. By computing the dot products with the individual terms (centroids and original query vector), the dot product calculation only needs to be done for the number of terms in the equation. Among those, the original query vector is very small and the dot product computation for this term is relatively inexpensive. In addition to this we have one (PsRF) or two (RF) dot products. We can then easily compute the dot products with the modified query vectors for different values of alpha, beta and gamma by a scalar addition with appropriate alpha, beta and gamma values. Thus the overall computation can be made faster.
The cosine similarity is obtained by dividing with the norms. This process is repeated for all the documents and the values of the cosine similarity metric for each query-document pair is stored.
9. For each query, the cosine similarity scores are sorted in descending order. The top 50 documents are then stored in a 2-column csv file in the format <query ID> : <document ID>. This is done for both RF and PsRF schemes separately with different values of alpha, beta and gamma.
10. After the ranked lists are generated, we then parse the data of generated Ranked list for all the 3 settings of alpha, beta and gamma, Gold standard ranked lists and Queries in an array
11. We maintained a document that maps the query id with the relevant documents and relevance score

12. We then compared which documents in our ranked list is present in the Gold Standard list and calculated the Precision@K and Average Precision.
13. The relevance of the documents is then stored in an array and compared to the sorted array of the relevance scores. This way, we calculate the NDCG.
14. We then average over all the queries to find out the average parameters.
15. After all the values of mAP@20 and NDGC@20 for all the 3 settings for RF and PsRF are calculated, the final values are put in a csv file.

Assumptions / Changes:

The term frequencies are not computed separately and are assumed to be stored in the inverted index itself.

Extra input / parameters:

The path to the "queries_<GROUP_NO>.txt" is to be given along with the path to the inverted index file, "en_BDNews24" folder , gold standard ranked list which is kept inside Data folder and has the name "rankedRelevantDocList.csv" and the "PAT2_<GROUP_NO>_ranked_list_A.csv" file. Here, GROUP_NO is 22.
The remaining files are intermediate files and will be generated in the root directory containing the .py file when the commands are run in serial order.

To run Task B1:

```
$>>python PB_22_rocchio.py <path to the en_BDNews24 folder> <path_to_model_queries_22.pth>
<path_to_gold_standard_ranked_list.csv> <path_to_PAT2_<GROUP_NO>_ranked_list_A.csv> <path to
queries_22.txt>

$>>python PB_22_evaluator.py ./Data/rankedRelevantDocList.csv PB_22_ranked_list_RF_0.csv
$>>python PB_22_evaluator.py ./Data/rankedRelevantDocList.csv PB_22_ranked_list_RF_1.csv
$>>python PB_22_evaluator.py ./Data/rankedRelevantDocList.csv PB_22_ranked_list_RF_2.csv

$>>python PB_22_evaluator.py ./Data/rankedRelevantDocList.csv PB_22_ranked_list_PsRF_0.csv
$>>python PB_22_evaluator.py ./Data/rankedRelevantDocList.csv PB_22_ranked_list_PsRF_1.csv
$>>python PB_22_evaluator.py ./Data/rankedRelevantDocList.csv PB_22_ranked_list_PsRF_2.csv

$>>python PB_22_Metrics_Generate.py PB_22_rocchio_RF_evaluated_0.csv
PB_22_rocchio_RF_evaluated_1.csv PB_22_rocchio_RF_evaluated_2.csv

$>>python PB_22_Metrics_Generate.py PB_22_rocchio_PsRF_evaluated_0.csv
PB_22_rocchio_PsRF_evaluated_1.csv PB_22_rocchio_PsRF_evaluated_2.csv
```

Python version used: 3.6.8

Library Requirements:

1. os
2. sys
3. pickle
4. numpy
5. math
6. csv
7. collections

Task B2 (Identifying words that are closer to the query)

Performed by:
Arghyadeep Bandyopadhyay (Roll No: 18EE10012)

Steps used:

The ranked list obtained after applying Relevance Feedback with alpha = 0.5, beta = 0.5 and gamma = 0.5 is used.

1. The paths and file names of all the documents are extracted from the 'en_BDNews24' folder.

2. The names of the top ten documents from the ranked list are stored for each query.

3. The inverted index file is read and the document frequencies (df) along with the vocabulary are stored.

4. For each document in the set of top ten retrieved documents over all queries, the document text is stored as a list of terms using the inverted index.

5. For each query, at first the |V|-dimensional normalised TF-IDF vectors for the top ten retrieved documents are obtained with the given weighting scheme (lnc.ltc). Then, the |V|-dimensional average of these 10 document vectors is obtained. The words corresponding to the 5 largest values in this vector are considered as the 5 words closest to this query. However, certain words namely: 'bdnews', 'com', 'reuters' and 'said' which were found to occur in most of the documents and were not related as such to any of the queries, were ignored while considering the 5 words closest to this query. The 5 largest values among the other words were considered. This process is carried out for all the queries.

6. For each query, the top 5 words are then stored in a 2-column csv file in the format <query ID> : <word1>, <word2>, <word3>, <word4>, <word5>.

Assumptions / Changes:

The term frequencies are not computed separately and are assumed to be stored in the inverted index itself. The words ‘bdnews’, ‘com’, ‘reuters’ and ‘said’ were not considered while obtaining the top 5 important words corresponding to a query.

To run Task B2:

```
$>>python3 PB_22_important_words.py  <path to the en_BDNews24 folder>
<path_to_model_queries_22.pth>  <path to ranked list obtained after applying
Relevance Feedback with alpha = 0.5, beta = 0.5 and gamma  = 0.5 >
```

Python version used: 3.6.8

Library Requirements:

1. os
2. sys
3. pickle
4. numpy
5. math
6. csv
7. collections