Group No: 10

Members:

- 18CS10037 Nikhil Popli
- 18CS10048 Sahil Jindal
- 18CS10052 Somay Chopra
- 18CS30009 Ashutosh Varshney

Task-2A

- Term frequencies are stored along with postings in the inverted index in task 1A which is used to find TF(t,d) efficiently. This is done by creating a 2D dictionary in the code where keys are the document ids and each value is a dictionary of term, term frequency.
- TF(t,q) is computed by simply using the saved parsed queries to get terms and corresponding term frequencies.
- For calculating IDF(t) first DF(t) is found which is simply the length of the postings list and subsequently appropriate function is applied to each term according to the scheme required. Since the scheme for getting IDF of query and document can be different, the aforementioned procedure is done separately for documents and queries.
- The vectors are formed by multiplying TF and IDF scores corresponding to each term and subsequently normalized.
- Scores are now found by taking the dot product of the query and document vector. Since the size of vocabulary is very large, this product is taken only over terms common to both document and query. This would work since all other terms would simply add up to 0 and won't contribute to the score.
- Finally, scores are sorted and the top 50 documents are saved corresponding to each query. This is done for all the 3 parts.
- The path to the parsed queries is expected as the 3rd argument while giving the command. The command to run is: python3 PAT2_10_ranker.py ./Data/en_BDNews24/ ./model_queries_10.pth ./queries_10.txt

Task-2B

- First the relevance scores for all the documents for each queries is obtained using the given rankedRelevantDocList.xslx file.
- For every query the DCG@10 and DCG@20 are found for the gold standard for ideal retrieval of relevant documents.
- Now for every output file (A/B/C) for each query AvgPrecision@10, AvgPrecision@20, NDCG@10 (DCG@10 for our model/DCG@10 forideal retrieval), NDCG@20(DCG@10 for our model/DCG@10 for ideal retrieval) are found.
- Finally the average of all the 4 matrices is taken over all queries.
- Commands: python PAT2_10_evaluator.py rankedRelevantDocList.xlsx PAT2_10_ranked_list_A.csv (Replace A with B and C for other files)