IRDM CW2 Report

1. Evaluating Retrieval Quality

I implemented 2 metrics for evaluating a ranking problem : 1. Average precision@k, 2. NDCG.

As the question asks, I acquired a ranked list from the validation data given using BM25 and computed the performance BM25.

1. Average precision@k

I implemented this metric in a fucntion called ‘average\_precision\_cal’ which takes a ranked list as a parameter.

I calculated the average precision of the ranked list by looping over the ranked list. During the loop, if a passage is a relevant passage, I calculate the precision of the list at that point and add to a variable called ‘precision\_sum’. For example, if the third passage is a relevant passage and it had 1 relevant passage on its previous steps, the precision at this point would be 2/3 and this will be added to ‘precision\_sum’.

When the loop is over, the average precision is calculated by dividing ‘precision\_sum’ by the length of the ranked list.

As for k, I used k = 100.

1. NDCG

I implemented this metric in a fucntion called ‘get\_NDCG’ which takes a ranked list as a parameter.

Firstly, I implemented a function which calculates the IDCG. To get IDCG value, I created a ideal ranked list from the ranked list by sorting the list by the ‘relevancy’ column. Then, by looping over the sorted list and using the equation below(Figure 1), I found the value of IDCG.

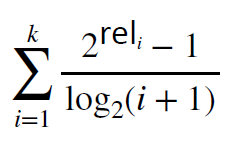


Figure 1

As for DCG, like what I did to get IDCG, I looped over the original ranked list and used the same equation to get DCG value. After the loop, I divided DCG value by IDCG value to get NDCG value.

1. Logistic Regression (LR)

How I dealt with imbalanced data and why I did it.

<https://towardsdatascience.com/how-i-handled-imbalanced-text-data-ba9b757ab1d8>

1. Inverted Index

The data structure I used to create an inverted index is Python dictionary. I believe Python dictionary is the most suitable data structure to use as it consist of a key and value pair. In my inverted index, keys are words in the passage and values are lists containing tuples that contain pid, word frequency, passage length in the form, (pid, word frequency, passage). For example, inverted\_index[‘definition’] will contain [(7130104, 3, 16), (8002085, 1, 39), …]. I stored not only the pid, but also the word frequency and the passage length to ease the calculation of TF-IDF weights. When calculating a term frequency for a document, the word frequency and passage length are needed because term frequency = frequency / passage length. I divide the frequency by the passage length to normalize the occurrence of the word with the size of the passage.

I created this inverted index by looping over the ‘cadidate\_passages\_top1000.tsv’ files.

1. Vector Space and BM25 Model

Conference Name:ACM Woodstock conference

Conference Short Name:WOODSTOCK’18

Conference Location:El Paso, Texas USA

ISBN:978-1-4503-0000-0/18/06

Year:2018

Date:June

Copyright Year:2018

Copyright Statement:rightsretained

DOI:10.1145/1234567890

RRH: F. Surname et al.

Price:$15.00