NLP Assignment 2

Sumanth Doddapaneni - CS21D409
Raghavan AK - CS21S025
Aswanth Kumar M - CS21M010

March 2022

1 Now that the Cranfield documents are preprocessed, our search engine needs a data structure to facilitate the 'matching' process of a query to its relevant documents. Let's work out a simple example. Consider the following three sentences:

S1 Herbivores are typically plant eaters and not meat eaters
S2 Carnivores are typically meat eaters and not plant eaters
S3 Deers eat grass and leaves

Assuming are, and, not as stop words, arrive at an inverted index representation for the above documents (treat each sentence as a separate document).

Terms	S1	S2	S3
herbivor	1	0	0
typic	1	1	0
plant	1	1	0
eater	2	2	0
meat	1	1	0
carnivor	0	1	0
deer	0	0	1
eat	0	0	1
grass	0	0	1
leav	0	0	1

2 Next, we must proceed on to finding a representation for the text documents. In the class, we saw about the TF-IDF measure. What would be the TF-IDF vector representations for the documents in the above table? State the formula used.

	Counts, tf						**tf * idf**		
Terms	**S1**	**S2**	**S3**	**df**	**D / df**	**idf**	**S1**	**S2**	**S3**
herbivor	1	0	0	1	3 / 1 = 3	0.477	0.477	0	0
typic	1	1	0	2	3 / 2 = 1.5	0.1760	0.1760	0.1760	0
plant	1	1	0	2	3 / 2 = 1.5	0.1760	0.1760	0.1760	0
eater	2	2	0	2	3 / 2 = 1.5	0.1760	0.352	0.352	0
meat	1	1	0	2	3 / 2 = 1.5	0.1760	0.1760	0.1760	0
carnivor	0	1	0	1	3 / 1 = 3	0.477	0	0.477	0
deer	0	0	1	1	3 / 1 = 3	0.477	0	0	0.477
eat	0	0	1	1	3 / 1 = 3	0.477	0	0	0.477
grass	0	0	1	1	3 / 1 = 3	0.477	0	0	0.477
leav	0	0	1	1	3 / 1 = 3	0.477	0	0	0.477

2.1 TF-IDF formula

TF-IDF(t, d) = TF(t, d) * IDF(t)
where ’t’ is term, ’d’ is document, TF is term frequency of term ’t’ in document ’d’ and IDF is inverse document frequency of term ’t’.

3 Suppose the query is ”plant eaters”, which documents would be retrieved based on the inverted index constructed before ?:

herbivore	S1 , S2
plant	S1 , S2
eater	S1 , S2
meat	S1 , S2
carnivore	S1 , S2
deer	S3
eat	S3
grass	S3
leav	S3

Retrived documents

S1 Herbivores are typically plant eaters and not meat eaters

S2 Carnivores are typically meat eaters and not plant eaters

4 Find the cosine similarity between the query and each of the retrieved documents. Rank them in descending order.

		Counts, tf							**tf * idf**		
Terms	**Q**	**S1**	**S2**	**S3**	**df**	**D / df**	**idf**	**Q**	**S1**	**S2**	**S3**
herbivore	0	1	0	0	1	3 / 1 = 3	0.477	0	0.477	0	0
typical	0	1	1	0	2	3 / 2 = 1.5	0.1760	0	0.1760	0.1760	0
plant	1	1	1	0	2	3 / 2 = 1.5	0.1760	0.1760	0.1760	0.1760	0
eater	1	2	2	0	2	3 / 2 = 1.5	0.1760	0.1760	0.352	0.352	0
meat	0	1	1	0	2	3 / 2 = 1.5	0.1760	0	0.1760	0.1760	0
carnivore	0	0	1	0	1	3 / 1 = 3	0.477	0	0	0.477	0
deer	0	0	0	1	1	3 / 1 = 3	0.477	0	0	0	0.477
eat	0	0	0	1	1	3 / 1 = 3	0.477	0	0	0	0.477
grass	0	0	0	1	1	3 / 1 = 3	0.477	0	0	0	0.477
leav	0	0	0	1	1	3 / 1 = 3	0.477	0	0	0	0.477

$|S_1| = \sqrt{0.4772^2 + 0.1760^2 + 0.1760^2 + 0.1760^2 + 0.352^2} = 0.66674582863$
$|S_2| = \sqrt{0.4772^2 + 0.1760^2 + 0.1760^2 + 0.1760^2 + 0.352^2} = 0.66674582863$
$|S_3| = \sqrt{0.4772^2 + 0.4772^2 + 0.4772^2 + 0.4772^2} = 0.95435842323$

$|Q| = \sqrt{0.1760^2 + 0.1760^2} = 0.2489$

$Q \cdot S_1 = 0.1760^2 + 0.1760 * 0.352 = 0.092928$

$Q \cdot S_2 = 0.1760^2 + 0.1760 * 0.352 = 0.092928$

$Q \cdot S_3 = 0$

$Cosine\Theta_{s_1} = \frac{Q \cdot S_1}{|Q| * |S_1|} = \frac{0.092928}{0.2489 * 0.66674582863} = 0.5599$

$Cosine\Theta_{s_2} = \frac{Q \cdot S_2}{|Q| * |S_2|} = \frac{0.092928}{0.2489 * 0.66674582863} = 0.5599$

$Cosine\Theta_{s_3} = \frac{Q \cdot S_3}{|Q| * |S_3|} = 0$

Ranking : S1, S2

5 Is the ranking given above the best?

No, The above results only produces documents which has exactly the words present in query. This results in S3 not being retrieved. But it should be retrieved as it has grass - close to plant , eater - close to eat. A better would be $S1 > S3 > S2$

7

7.1 What is the IDF of a term that occurs in every document?

The IDF of a term that occurs in every document is 0

7.2 Is the IDF of a term always finite? If not, how can the formula for IDF be modified to make it finite?

The IDF value is not always finite, when the term is not present in corpus, but present in query, this will lead to a division-by-zero. It is therefore common to smooth by adding a constant to the denominator. Adjusted definition is

IDF = $\frac{N}{d+1}$

where N is total number of documents, and d is no of documents where the term appear.

8 Can you think of any other similarity/distance measure that can be used to compare vectors other than cosine similarity. Justify why it is a better or worse choice than cosine similarity for IR.

Manhattan Distance: Manhattan distance is a distance metric between two points in a N dimensional vector space. It is the sum of the lengths of the projections of the line segment between the points onto the coordinate axes.

Euclidean Distance: The Euclidean distance corresponds to the L2-norm of a difference between vectors, gives a sense of how closer vectors are in terms of their distances. However this metric is not scale invariant. The dense vectors result in large values which marks it hard to set scales.

Jaccard Similarity: Jaccard Similarity measures similarities between sets. It is defined as the size of the intersection divided by the size of the union of two sets. The problem in such approach is that we are again relying on the frequency counts and not taking IDF of terms into account.

The cosine similarity is advantageous because even if the two similar documents are far apart by the Euclidean distance (due to the size of the document), chances are they may still be oriented closer together. The smaller the angle, higher the cosine similarity.

9 Why is accuracy not used as a metric to evaluate information retrieval systems?

There is a good reason why accuracy is not an appropriate measure for information retrieval problems. In almost all documents to be searched, the data is extremely skewed: normally over 99.9% of the documents are in the nonrelevant category. A search system tuned to maximize accuracy can appear to perform well by simply deeming all documents nonrelevant to all queries. Even if the system is quite good, trying to label some documents as relevant will almost always lead to a high rate of false positives. However, labeling all documents as nonrelevant is completely unsatisfying to an information retrieval system user. Users are always going to want to see some documents, and can be assumed to have a certain tolerance for seeing some false positives providing that they get some useful information. The measures of precision and recall concentrate the evaluation on the return of true positives, asking what percentage of the relevant documents have been found and how many false positives have also been returned.

10 For what values of α does the F_α -measure give more weightage to recall than to precision?

F_α is defined as :

$$F_\alpha = \frac{1}{\frac{\alpha}{precision} + \frac{1-\alpha}{recall}}$$

1. For $\alpha = 1$, $F_\alpha = Precision$

2. For $\alpha = 0$, $F_\alpha = Recall$

3. For $\alpha \in (0.5, 1]$, the F_α score gives more weightage to precision than recall.

11 What is a shortcoming of Precision @ k metric that is addressed by Average Precision @ k?

Precision @ k metric fails to take into account the positions of the relevant documents among the top k. Where as, Average precision @ k considers the order in which the returned documents are presented. [1]

For example, consider the case where documents retrieved by two IR systems [1, 1, 1, 0, 0, 0, 0, 0, 0, 0] and [0, 0, 0, 0, 0, 0, 0, 1, 1, 1] (where 1 implies that document is relevant and 0 as not relevant). On computing the precision @ 10 score, both the IRs give the same value but the first IR gives a better ranking of relevant documents compared to second IR. In case of Average Precision @ 10, the score of first IR will be higher compared to second IR as it considers the order of the relevant documents.

Formula for Average Precision @ k: AveP @ k = $\frac{\sum_{k=1}^{n} P(k) \times rel(k)}{\text{number of relevant documents}}$

12 What is Mean Average Precision (MAP) @ k? How is it different from Average Precision (AP) @ k?

Mean Average Precision(MAP) for a set of queries is the mean of the average recision scores for each query [1].

$$\text{MAP @ k} = \frac{\sum_{q=1}^{Q} AveP(q)}{Q}$$

Where Q is the number of quries

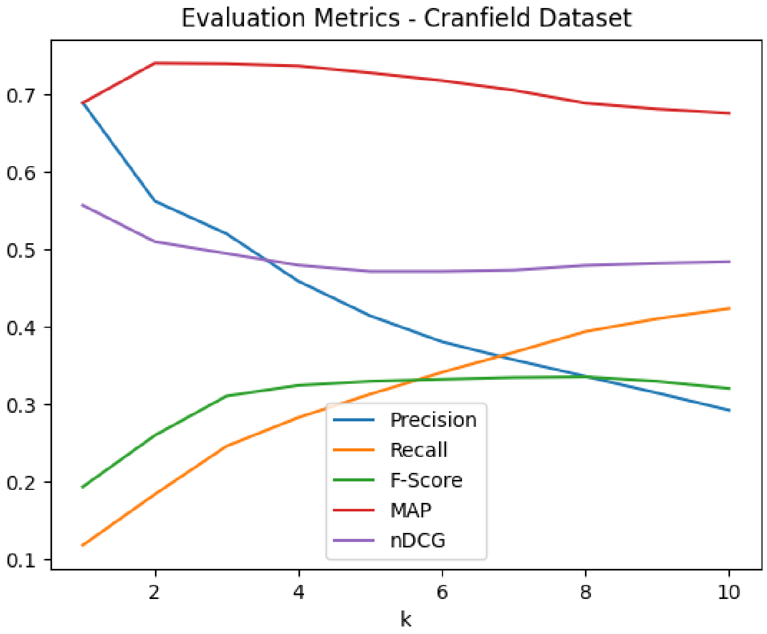
AP @ k is defined only for one query where as MAP @ k is measured for a set of quries. So, MAP @ k is a better metric for an IR system.

13 For Cranfield dataset, which of the following two evaluation measures is more appropriate and why? (a) AP (b) nDCG

For Cranfield dataset, **nDCG** is a more appropriate evaluation measure compared to **AP**.
The reasons are [2]:

1. The nDCG takes into account the graded relevance values, where as in AP it is either binary (relevant/non-relevant).
2. nDCG is a good metric at evaluating the position of ranked items.
3. As, nDCG operates beyond the binary relevant/non-relevant scenario.

15 Assume that for a given query, the set of relevant documents is as listed in cranqrels.json. Any document with a relevance score of 1 to 4 is considered as relevant. For each query in the Cranfield dataset, find the Precision, Recall, F-score, Average Precision and nDCG scores for k = 1 to 10. Average each measure over all queries and plot it as function of k. Code for plotting is part of the given template. You are expected to use the same. Report the graph with your observations based on it.



Following are the observation from the plot:

1. With precision decreasing proportional to rank shows that the IR system is good. It shows that all relevant data docs are being retrieved at the top positions
2. Recall can be seen to increase monotonically increasing with k, which is

also expected as more and more documents are retrieved

3. F-score is the harmonic mean of precision and recall. Based on the graphs of precision and recall graphs, f-score initially increases with k and the graph becomes flat with increasing k.

4. Mean Average Precision can be seen to be increasing first, reaching a maxima and then starts decreasing. It is the average across queries and it's a good sign the most relevant docs are found in the top ranks

5. NDCG is a measure of ranking quality. Note that in a perfect ranking algorithm, the DCG will be the same as the IDCG producing an nDCG of 1.0. All nDCG calculations are then relative values on the interval 0.0 to 1.0 and so are cross-query comparable. The nDCG values seem to be relatively constant with a minor increase for larger values of k. [2]

16 Analyse the results of your search engine. Are there some queries for which the search engine's performance is not as expected? Report your observations.

Some of the queries where the search engine's performance is not as expected are:

1. Query number: 63, with query: "where can i find pressure data on surfaces of swept cylinders ." . The relevant documents for this query are: 539, 567, 564, 566 .
   The IR system retrieved these documents as top 10 documents: 738, 1045, 839, 843, 1046, 678, 891, 491, 1051, 1121. We can see that in the retrieved documents there is no document that is actually relevant to the query.

2. Query number: 85, with query: "what parameters can seriously influence natural transition from laminar to turbulent flow on a model in a wind tunnel ." . The relevant documents for this query are: 546, 608, 406, 606, 710 .
   The IR system retrieved these documents as top 10 documents: 294, 418, 315, 295, 1155, 272, 522, 1153, 431, 207. We can see that the IR system fails to retrieve even a single actual relevant document among 10 predicted documents.

3. Similar to the above queries, query number: 117, with query: "is there any information on how the addition of a /boat-tail/ affects the normal force on the body of various angles of incidence ." also has same issue.

   All these queries match key words in the query, words like *pressure, cylinder, etc.* and return wrong documents. Vector space models suffer from these keyword matching problems.

17 Do you find any shortcoming(s) in using a Vector Space Model for IR? If yes, report them.

Some short comings of vector space model for IR are: [3]

1. Since each word is a dimension, consideration of all words is impractical.
2. Considering all words would imply expensive computation in a very high dimensional space.
3. Also, it assumes all words are independent which is not true.

18 While working with the Cranfield dataset, we ignored the titles of the documents. But, titles can sometimes be extremely informative in information retrieval, sometimes even more than the body. State a way to include the title while representing the document as a vector. What if we want to weigh the contribution of the title three times that of the document?

Document titles can be merged into the document body and we can give higher weightage to document titles during the computation of term frequency. For example, we can assume that the document title contributes 3 times the TF-IDF representation of the document title.

$$\text{TF} = 1\times\text{number of times word is in the body}+3\times\text{number of times word is in body}$$

(where TF is term frequency.)

We need to ensure that the document title contribution is neither high nor low magnitude. There has to be a right balance of weighted representation of document title.

19 Suppose we use bigrams instead of unigrams to index the documents, what would be its advantage(s) and/or disadvantage(s)?

The advantages of using bigrams instead of unigrams are [4]:

1. Bigrams provide a better context than Unigram. Example: "White house" is a good indication to politics, when used unigrams then both "White" and "house" are less informative.
2. Bigrams are better at modelling the sequences as it considers two words at a time which gives a better insight on order of words.
3. As bigrams are better at context and sequence modelling, it has a better precision than unigrams.

The disadvantages of using bigrams instead of unigrams are [4]:

1. The number of occurences of each bigram will be lower (compared to unigrams) so computation will become more sensitive to noise.
2. The vector space formed by bigrams is very high dimensional as the number of bigrams formed are more. This leads to sparse vector representation and require more computational time.
3. The recall is lower compared to unigrams.

20 In the Cranfield dataset, we have relevance judgements given by the domain experts. In the absence of such relevance judgements, can you think of a way in which we can get relevance feedback from the user himself/herself? Ideally, we would like to keep the feedback process to be non-intrusive to the user. Hence, think of an 'implicit' way of recording feedback from the users.

The relevance feedback from the user can be obtained by[5]:

1. Noting which documents they do and do not select for viewing
2. Duration of time spent on viewing a document
3. Page browsing or scrolling actions with in a document
4. User's Browsing history
5. User's search or query history
6. Per User time spent

On considering the above parameters in combination and on compiling the same query across different users, we can conclude which documents are relevant for a query.

References

[1] https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval).

[2] https://medium.com/swlh/rank-aware-recsys-evaluation-metrics-5191bba16832.

[3] https://www.sciencedirect.com/topics/computer-science/vector-space-models.

[4] https://stats.stackexchange.com/questions/231427/changing-classifier-input-features-from-unigrams-to-bigrams.

[5] https://en.wikipedia.org/wiki/Relevance_feedback.