Natural Language Processing (CS6370)
Assignment 2

March 20, 2023

Team Number 42
Pranjul Dubey (ED21S011)
Bhumika Gupta (EE21S060)

---

1. Now that the Cranfield documents are pre-processed, 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).

   Ans: After tokenisation and stopword removal the documents look like:
   S1 herbivores, typically, plant, eaters, meat, eaters
   S2 carnivores, typically, meat, eaters, plant, eaters
   S3 deers, eat, grass, leaves

   Inverted Indexing:
   $herbivores \rightarrow S1$
   $typically \rightarrow S1, S2$
   $plant \rightarrow S1, S2$
   $eaters \rightarrow S1, S2$
   $meat \rightarrow S1, S2$
   $carnivores \rightarrow S2$
   $deers \rightarrow S3$
   $eat \rightarrow S3$
   $grass \rightarrow S3$
   $leaves \rightarrow S3$

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.

Ans: TF-IDF for the above documents by considering the formulae as:

(a) Term Frequency, tf(t,d), of term(t) in document(d) is the relative frequency of term t within document d and is given as:

$$tf(t,d) = \text{Freq. of term(t) in document(d)}$$

Note: The denominator is calculated by considering each occurrence of the same term separately.

(b) Inverse Document Frequency, idf(t,D), measures how much information a particular word provides. It is calculated for the whole collection of documents. It is calculated as:

$$idf(t,d) = log(\frac{\text{Total Number of Documents in corpus (D)}}{\text{Total Number of documents in which term (t) occurrs}})$$

(c) TF-IDF: After calculating TF and IDF using above formulae, we calculate TF-IDF by multiplying these together. i.e.,
$\text{tfidf}(t,d,D) = \text{tf}(t,d) \cdot \text{idf}(t,D)$

Total number of documents in given corpus (D) = 3
IDF table :

Term (t)	**Documents containing (t)**	**IDF**
herbivores	1	$\log(\frac{3}{1}) = 0.477$
typically	2	$\log(\frac{3}{2}) = 0.176$
plant	2	$\log(\frac{3}{2}) = 0.176$
eaters	2	$\log(\frac{3}{2}) = 0.176$
meat	2	$\log(\frac{3}{2}) = 0.176$
carnivores	1	$\log(\frac{3}{1}) = 0.477$
deers	1	$\log(\frac{3}{1}) = 0.477$
eat	1	$\log(\frac{3}{1}) = 0.477$
grass	1	$\log(\frac{3}{1}) = 0.477$
leaves	1	$\log(\frac{3}{1}) = 0.477$

TF table:

Doc	herbivores	typically	plant	eaters	meat	carnivores	deers	eat	grass	leaves
S1	1	1	1	2	1	0	0	0	0	0
S2	0	1	1	2	1	1	0	0	0	0
S3	0	0	0	0	0	0	1	1	1	1
query	0	0	1	1	0	0	0	0	0	0

Now, we can calculate TF-IDF values which is given in the following TF-IDF table:

Doc	herbivores	typically	plant	eaters	meat	carnivores	deers	eat	grass	leaves
S1	0.477	0.176	0.176	0.352	0.176	0	0	0	0	0
S2	0	0.176	0.176	0.352	0.176	0.477	0	0	0	0
S3	0	0	0	0	0	0	0.477	0.477	0.477	0.477
query	0	0	0.176	0.176	0	0	0	0	0	0

The row-wise entries can be read as the TF-IDF vector representations of the given documents.

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

Ans: Given query upon tokenization:
Q $\rightarrow$ *plant, eaters*
From the inverted indexing obtained in answer 1, we can see that the two documents *S1, S2* will be retrieved.

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

Ans: Cosine similarity between a document and a query is given as:

$$sim(d_i, q) = \frac{d_i \cdot q}{\|d_i\|\|q\|}$$

For the given documents,
$\|d_1\| = \sqrt{(0.477)^2 + (0.176)^2 + (0.176)^2 + (0.352)^2 + (0.176)^2} = 0.667$
$\|d_2\| = \sqrt{(0.176)^2 + (0.176)^2 + (0.352)^2 + (0.176)^2 + (0.477)^2} = 0.667$
$\|d_3\| = \sqrt{(0.477)^2 + (0.477)^2 + (0.477)^2 + (0.477)^2} = 0.894$

$\|q\| = \sqrt{(0.176)^2 + (0.176)^2} = 0.249$

$d_1 \cdot q = 0.176 * 0.176 + 0.352 * 0.176 = 0.0929$
$d_2 \cdot q = 0.176 * 0.176 + 0.352 * 0.176 = 0.0929$

$sim(d_1, q) = \frac{d_1 \cdot q}{\|d_1\|\|q\|} = \frac{0.0929}{0.667 * 0.249} = 0.559$
$sim(d_2, q) = \frac{d_2 \cdot q}{\|d_2\|\|q\|} = \frac{0.0929}{0.667 * 0.249} = 0.559$

5. Is the ranking given above the best?

Ans: No, the ranking obtained above is not the best for the reasons stated below:

(a) Dispute in the ranking if the similarity score is same.

(b) Non-retrieval of the document that could be relevant, thereby affecting recall.
As we can see that for the given query, document 3 (i.e., S3) is potentially relevant (as the query is about "plant eaters" and S3 mentions "deers", "grass" and "leaves" which are closely related to "plant") but since it doesn't contain the exact words as in the query, it is marked as irrelevant hence, not retrieved.

(c) Retrieval of the irrelevant document just because it contains the query terms, thereby affecting precision.
As we can observe, for the given query, document 2 (i.e., S2) is potentially irrelevant (as the query is about "plant eaters" whereas S2 talks about "carnivores") but is still marked as relevant.

6. Now, you are set to build a real-world retrieval system. Implement an Information Retrieval System for the Cranfield Dataset using the Vector Space Model.

Ans: Implemented in code.

7. (a) What is the IDF of a term that occurs in every document?

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

Ans:

(a) The IDF of a term that occurs in every document = $log(D/D) = log(1) = 0$ where, D = Total number of documents

(b) No, if there exists a term in query which does not occur in the given document collection then $idf(t, D) = log(D/0) = \infty$
For making the idf finite, we do smoothing as follows:

$$idf(t, d) = log(\frac{D+1}{df(t)+1})$$

where,
D = Total Number of Documents in corpus
df(t) = Total Number of documents in which term (t) occurrs

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.

Ans: Yes, there are several similarity/distance measures that can be used to compare vectors, including Euclidean distance, Jaccard similarity, and Pearson correlation coefficient.

(a) Euclidean distance: It measures the straight-line distance between two points in n-dimensional space. It is a simple and intuitive measure of similarity, but it may not always be the best choice for information retrieval (IR) tasks as the document vectors generally have a higher magnitude than the query vector.

(b) Manhattan distance: It measures the distance between two vectors by summing the absolute differences between their corresponding coordinates. It can be useful when dealing with sparse vectors or when the magnitude of the differences between coordinates is more important than their direction which is usually not the case in IR.

(c) Pearson correlation coefficient: It measures the linear correlation between two vectors. It can be a useful similarity measure when dealing with continuous data and when the direction and magnitude of differences between coordinates are both important which again is not the case in IR.

(d) Jaccard similarity: It measures the similarity between two sets by comparing the number of common elements to the total number of distinct elements in the two

sets. It can be a better choice than cosine similarity in IR when focusing on the presence or absence of specific features rather than their frequency or magnitude. It does not consider the magnitude of the sets being compared, which can lead to issues when dealing with very small or very large sets.

Cosine similarity is a popular choice for information retrieval because it is effective in capturing the similarity between vectors regardless of their magnitudes. It also has a simple and computationally efficient formula allowing fast calculations on large datasets.

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

Ans:

	Relevant	Nonrelevant
Retrieved	true positives (t_p)	false positives (f_p)
Not retrieved	false negatives (f_n)	true negatives (t_n)

In terms of the contingency table above, Accuracy can be written as:
Accuracy = $\frac{(t_p+t_n)}{(t_p+f_p+f_n+t_n)}$

It is not an appropriate measure for information retrieval problems because, generally the data is extremely skewed, i.e., normally, over 99.9% of the documents are in the nonrelevant category. This means that if a system is designed to maximize accuracy, then it will simply tend to perform well by labeling all the documents as nonrelevant to all the queries, which is highly undesirable.

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

Ans: $F_\alpha = \frac{(\alpha^2+1)PR}{(\alpha^2 P+R)}$
where,
P = Precision
R = Recall

For the values of α ¿ 1, the F_α measure gives more weightage to recall than precision.

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

Ans:

(a) Precision:
It quantifies the number of relevant items in the top-K retrieved results. It is given as:
Precision@k = $\frac{truepositives@k}{truepositives@k + falsepositives@k}$

(b) Average Precision:
It evaluates whether all of the ground-truth relevant items selected by the model are ranked higher or not.
AP @k = $\frac{\sum_{k=1}^{n}(P(k) * rel(k))}{\text{number of relevant items}}$
where,
rel(k) is an indicator function which is 1 when the item at rank k is relevant.
P(k) is the Precision@k metric

Precision @ k metric only considers the precision of the top k results, without taking into account the order of the results beyond the top k.

In contrast, the Average Precision @ k metric takes into account the relevance of all retrieved results up to rank k. It computes the average precision at each rank where a relevant document is found, and then averages these values over all relevant documents. This means that Average Precision @ k is a more comprehensive measure of retrieval performance that accounts for the relevance of all retrieved documents, not just the top k.

Therefore, Average Precision @ k is particularly useful when the goal is to retrieve all relevant documents within the top k results, or when the ranking of the documents beyond the top k is important for the user.

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

Ans: Mean Average Precision (MAP) @ k is a widely used evaluation metric in information retrieval that provides an overall measure of the effectiveness of a ranking algorithm at retrieving relevant documents. It is the **average of the Average Precision (AP) scores for each query** in a test collection. Mathematically, it is

written as:
$MAP = \frac{1}{Q} \sum_{q=1}^{Q} AP(q)$

The AP @ k is a metric that measures the precision of the retrieved documents up to a certain rank k. It computes the average precision at each rank where a relevant document is found, and then averages these values over all relevant documents. The **AP @ k is computed separately for each query** in the test collection.

The key difference between MAP @ k and AP @ k is that AP @ k calculates the average precision at a given rank k for a single query, whereas MAP @ k calculates the average of the AP @ k values across multiple queries. AP @ k is useful when evaluating the performance of a system for a specific query, whereas MAP @ k is useful when evaluating the performance of a system across a set of queries.

MAP @ k is particularly useful when comparing the performance of different ranking algorithms or when optimizing the performance of a single algorithm over multiple queries. By taking into account the AP scores for all queries, it provides a more robust measure of the effectiveness of a ranking algorithm on a collection of queries.

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

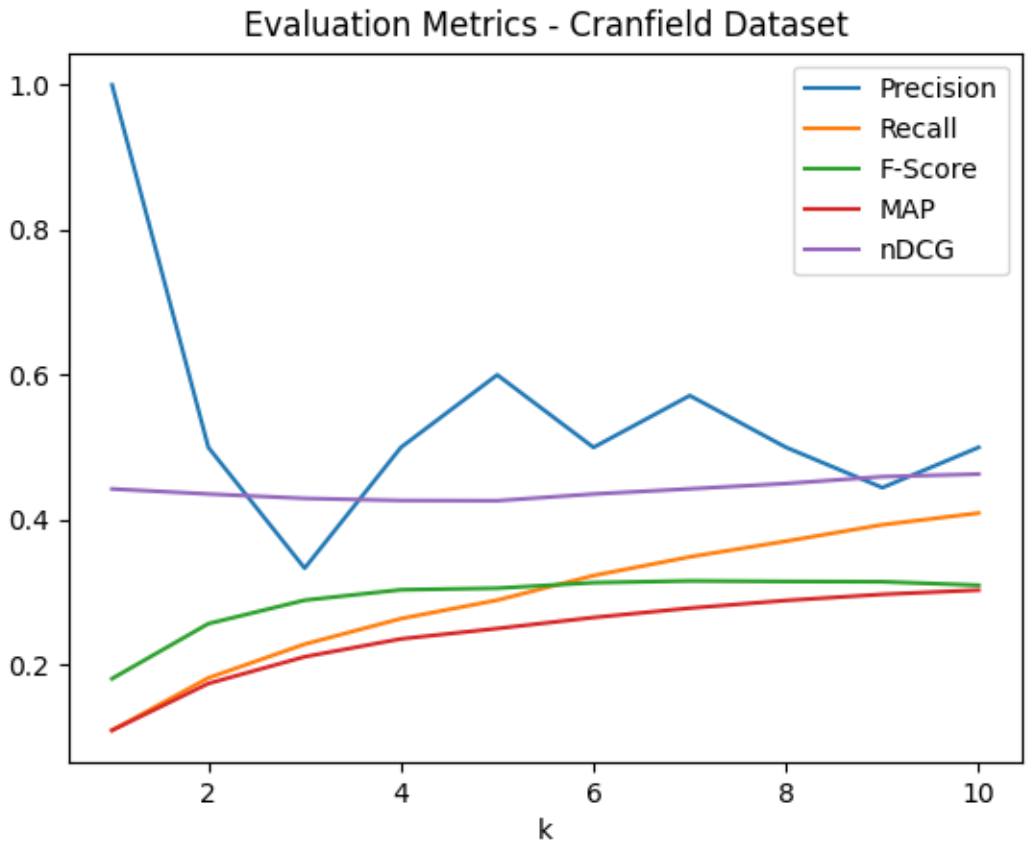
Ans: As the Cranfield dataset contains the non binary relevance ranking (1-4) for the queries, we should use nDCG measure where the goal is to rank documents based on their relevance and to ensure that the top-ranked documents are the most relevant ones which in general is the prime motive of IR systems. The nDCG thus can be averaged for all queries to get the overall performance measure of the IR. The AP is calculated on the basis of a single query and may not be as useful to evaluate the IR system's overall performance.

14. Implement the following evaluation metrics for the IR system:
    (a) Precision @k
    (b) Recall @k
    (c) F-Score @k
    (d) Average Precision @k
    (e) nDCG @k

Ans: Implemented in code.

15. Assume that for a given query, the set of relevant documents is as listed in cran_qrels.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.

Ans: The obtained graph is as follows:



Evaluation Metrics
Following observations can be made:

(a) Precision: decreases with k

(b) Recall: non-decreasing function with respect to k

(c) F1-Score : increases then stabilises as it depends on both precision and recall. F1 measures gives equal weightage to precision and recall.

(d) MAP: Gives a single metric that represents the complex Area under the Precision-Recall curve.

(e) nDCG : takes into account the graded relevance values as we have in the given dataset .

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.

Ans: An exemplar query for which the IR system doesn't perform well is:
query number 9 :"papers on internal /slip flow/ heat transfer studies ."

Reason could be because of improper use of punctuation so the term fail to match the type in vocabulary In contrast, humans can easily interpret it as equivalent to slip, flow , slip-flow.

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

Ans: Disadvantages:

(a) Assumes the terms to be independent. Hence, if the query phrase contains the term "love" it will only match the documents containing "love" and leave the documents containing "affection".

(b) Calculation intensive

(c) Update is not easy. Requires idf calculation all over again.

(d) Lengthy documents get poor similarity scores even though highly relevant.

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?

Ans: One possible way of implementing this could be:
Taking all the types from titles as well as contents of all the documents and representing the titles and contents in this vector space. We can then scale the title vectors by multiplying it by 3 and then adding to the corresponding content vectors. This way, we can ensure that we are weighing the contribution of titles three times as that of the document.

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

Ans: Advantages:

(a) Captures more context: bigrams can capture more context than unigrams as it takes into account pairs of adjacent words that may convey more meaning than when they are separated by other words. For example, the bigram "machine learning" carries more specific meaning than the unigrams "machine" and "learning" individually.

(b) Improved precision: using bigrams can help reduce ambiguity and improve precision in information retrieval. For example, the bigram "data science" can more precisely identify documents that are specifically about data science, whereas the unigrams "data" and "science" may appear in many irrelevant documents.

Disadvantages:

(a) Increased index size: by using bi-grams instead of uni-grams, the index size will be larger, which can be problematic for storage and processing, hence can make system slow.

(b) Increased sparsity: the larger index size can also result in more sparsity in the index, which can negatively affect retrieval performance.

(c) More complex queries: using bi-grams can make query formulation more complex. In order to match bigrams, queries need to include two or more terms, which can be challenging for users who are not familiar with the specific terminology.

(d) Reduced recall: queries with only single term may no longer match to the documents unless is a huge disadvantage in recall centric IRs

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.

Ans:

The various kinds of Implicit relevance feedback (IF) that can be obtained are:

(a) Clickthrough data: A commonly used form of implicit feedback in which observations are made of the search results that a user clicks on. The idea is that the user is more likely to click on relevant results.

(b) User's query history: Provides valuable insights into user's search behavior. This includes observing query rewrites, which indicate dissatisfaction with the initial search results, and examining previous queries to disambiguate potentially ambiguous queries. For example, knowing that a user recently searched for "C++" could help clarify their intent when searching for "Java" which has multiple meanings.

(c) User's entire history: Web pages, emails, calendar items, and documents in their filesystem, can be observed to infer relevance judgements according to Teevan et al. [TDH05].

(d) Reading time: Observation of reading time on search results may indicate relevance, but its correlation with user interest is still under debate.

(e) Eye tracking: A method of observing user behavior by measuring features such as eye fixation and pupil dilation. The hypothesis is that relevant and non-relevant results can be distinguished by differences in these features, such as larger pupil diameter being an indication of relevance. Saloj¨arvi et al. conducted a small study to measure these effects.

REFERENCES

- http://www.minerazzi.com/tutorials/term-vector-3.pdf
- https://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-unranked-retrieval-sets-1.html
- https://amitness.com/2020/08/information-retrieval-evaluation/
- https://www.cs.cornell.edu/courses/cs6740/2010sp/guides/lec15.pdf
- https://redirect.cs.umbc.edu/ ian/irF02/lectures/07Models-VSM.pdf