CS6370: Assignment I - Part 2

Shashank M Patil
CH18B022

Shania Mitra
CH18B067

6 April 2021

Jan-May 2021 CS6370: Natural Language Processing
Release Date: 16/03/2021
Deadline: 6/04/2021

The goal of the assignment is to build a search engine from scratch, which is an example of an information retrieval system. In the class, we have seen the various modules that serve as the building blocks of a search engine. The first part of the assignment involved building a basic text processing module that implements sentence segmentation, tokenization, stemming/lemmatization and stopword removal. This module involves implementing an Information Retrieval system using the Vector Space Model. The same dataset as in Part 1 (Cranfield dataset) will be used for this purpose.

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).

   Answer:
   After initial preprocessing (spellcheck, lemmatization and stopword removal) we have inverted index representation as follows

Terms	Documents
herbivore	$D1$
typically	$D1, D2$
plant	$D1, D2$
eater	$D1, D2$
meat	$D1, D2$
carnivore	$D2$
deer	$D3$
eat	$D3$
grass	$D3$
leaf	$D3$

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.

Answer:
For a given term,
TF = number of times the term appears in the document/Total number of terms in that document
The reason we normalize is that some documents have a large number of terms while some have a small number of terms. A term that occurs five times in a document with 10 words must have more weight than one which occurs five times in a document with 1000 words. Hence, using proportion would be more meaningful.
IDF = $log(N/n)$ where N = Total number of documents, n = Number of documents the term appears in

TF-IDF representation where each cell is $TF * IDF$ of that term

Document	herbivore	typically	plant	eater	meat	carnivore	deer	eat	grass	leaf
$D1$	0.096	0.036	0.036	0.07	0.036	0	0	0	0	0
$D2$	0	0.036	0.036	0.07	0.036	0.096	0	0	0	0
$D3$	0	0	0	0	0	0	0.12	0.12	0.12	0.12

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

Answer:
'plant' is present in $D1, D2$ and 'eater' (after lemmatization) in $D1, D2$
Therefore, the retrieved documents would be $D1, D2$

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

Answer:

TF-IDF representation of the query,
TF: plant = 0.5 , eater = 0.5
IDF: plant = 0.18, eater = 0.18

Query	herbivore	typically	plant	eater	meat	carnivore	deer	eat	grass	leaf
Q	0	0	0.09	0.09	0	0	0	0	0	0

To calculate TF-IDF of the query, we use the term frequency of the word in the query and the pre-existing IDF value of the words as calculated from the documents
Cosine similarity between document, D and query, Q can be calculated as,

$$Sim(Q,D) = cos(\theta) = \frac{\mathbf{Q} \cdot \mathbf{D}}{\|\mathbf{Q}\|\|\mathbf{D}\|}$$

$$cos(\theta_{D1}) = \left(\frac{(0.09*0.036)+(0.09*0.07)}{\sqrt{(0.09^2+0.09^2)}\sqrt{(0.096^2+0.036^2+0.036^2+0.07^2+0.036^2)}} \right) = 0.56$$
$$cos(\theta_{D2}) = \left(\frac{(0.09*0.036)+(0.09*0.07)}{\sqrt{(0.09^2+0.09^2)}\sqrt{(0.036^2+0.036^2+0.07^2+0.036^2+0.096^2)}} \right) = 0.56$$
$$cos(\theta_{D3}) = 0$$

Therefore the ranking would be,
Rank-1 = $D1, D2$
Rank-2 = $D3$

5) Is the ranking given above the best?

Answer:
We expect $D3$ to be ranked above $D2$ for the given query, since for the query 'plant eater', we would expect herbivore related documents but using the above model we are even getting carnivore related document. Therefore, the above ranking is not the best since we expect the result to be $D3$ but the returned results are $D1$ and $D2$.

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.
Answer:
Refer to the code for implementation

7) (a) What is the IDF of a term that occurs in every document?
Answer:
The IDF of such a term is 0 since n = N, $log(N/N) = log(1) = 0$

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

Answer:
No, the IDF need not always be finite. If there is a term in the query which does not occur in any of the documents,
$n = 0$
$\implies log(N/n) = \infty$

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.

Answer:

Other possible measures that could be used are:

- **Euclidean Distance**: This performs worse since the absolute values of the elements in the vectors are of little importance here. Typically the query vector has fewer words than document vectors leading to large euclidean distance of the queries with each of the documents. Vectors that are 'far apart' in terms of distance but parallel(having similar proportions of words) may be much more similar than vectors having similar magnitudes for elements, but in a different direction.
- **Jaccard similarity:** This measure treats the elements of the vector as an unordered set.

  $$J(A,B) = \frac{\mid A \cup B \mid}{\mid A \cap B \mid}$$

  This performs worse again, since a query typically has fewer words than a document and hence the intersection of elements is likely to be ϕ.

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

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Accuracy is not used as a metric to evaluate IR systems because TN values (The number of documents that are not relevant and not returned) is very large. Thus, it leads to accuracy having a high value even for IR systems that perform poorly. Further it is not sensitive to changes in FP and FN values since they are small and are overshadowed by the large TN value. This makes accuracy an uninformative and unimportant choice as an IR evaluation metric.

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

Answer:

$$F_\alpha = \left(\frac{1}{\frac{\alpha}{P} + \frac{1-\alpha}{R}} \right)$$

To give more weightage to recall than to precision we should have $\alpha < 0.5$

11) What is a shortcoming of Precision @ K metric that is addressed by Average Precision @ k?
Answer:
Precision@k would consider the performance of the IR system at that particular k, irrespective of the order the relevant documents occur in among the top-k results. However, in an IR system the ranks of documents are also extremely important since we would like to see the most relevant documents at the top. For example say for a particular query, system 1 outputs the 3 most relevant documents at the top and 3 irrelevant documents at the bottom, while system 2 outputs 3 irrelevant documents at the top and 3 relevant documents at the bottom. Even though we understand that system 1 does better since the most relevant ones are at the top, we get the same value for precision@6. Average Precision @ 6 addresses this by calculating the precision at all ranks in which relevant documents occur. Thus for system 1, the average precision is calculated at rank 1, 2 and 3 and comes out to be 1, while average precision for system 2 is calculated at ranks 4, 5, 6 and comes out to be 0.383, clearly indicating system 1 to be better than system 2, thus accounting for the ranking of the output.

12) What is Mean Average Precision (MAP) @ k? How is it different from Average Precision (AP) @ k ?
Answer:
Mean Average Precision is average precision at k, averaged over all the queries. Even though the terms mean and average mean the same, here they have completely different meanings. Average refers to averaging over the ranks at which relevant documents occur, while mean

refers to averaging over all the queries. Hence, average precision is for one query while MAP is for AP for multiple queries. The two would be the same if only one query is considered.

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

Answer:
Average Precision considers only binary degrees of relevance while nDCG considers varying degree of relevances. Since degrees of relevance are given to us, nDCG would be more appropriate.

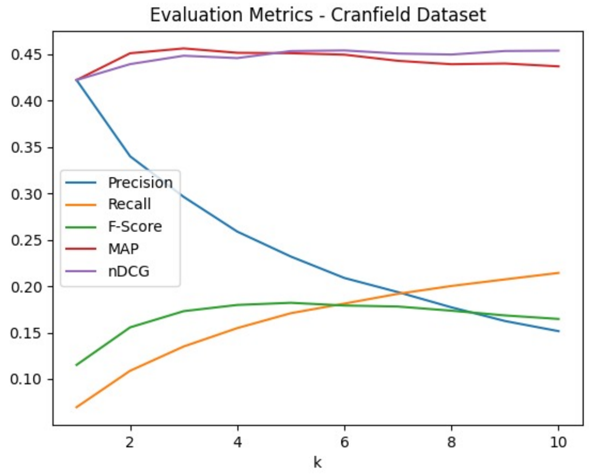
14) Implement the following evaluation metrics for the IR system:

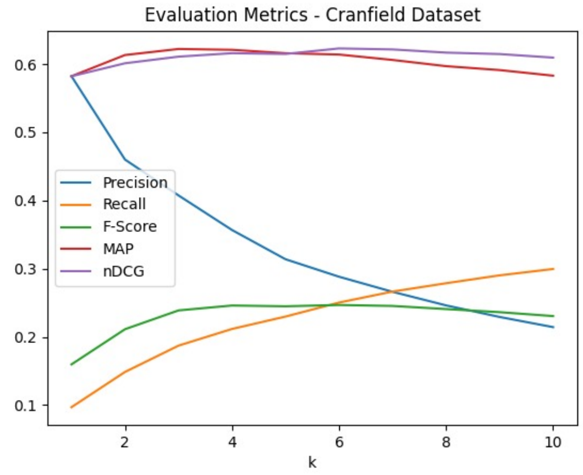
  a) Precision @ k
  b) Recall @ k
  c) F-Score @ k
  d) Average Precision @ k
  e) nDCG @ k

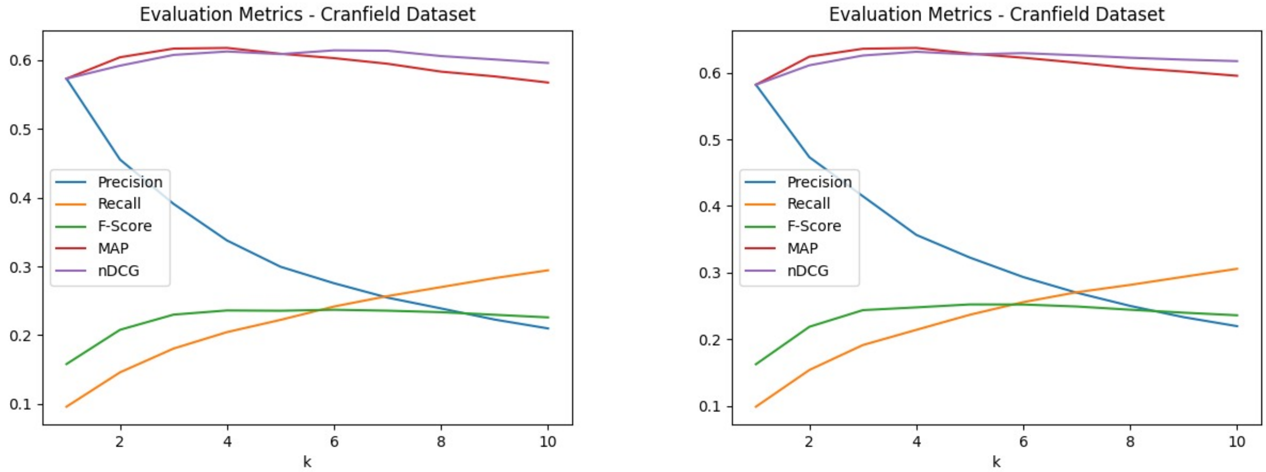
  Answer:

  Refer to the code for implementation

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.**







Top Left - *Segmenter* : Naive ,*Tokenizer* : Penn Treebank
Top Right - *Segmenter* : Naive ,*Tokenizer* : Naive
Bottom Left - *Segmenter* : Punkt ,*Tokenizer* : Naive
Bottom Right- *Segmenter* : Punkt ,*Tokenizer* : Penn Treebank

- The general trend across all the graphs is the same. We see that the IR system has a monotonically decreasing mean precision (averaged over all queries) and as expected a monotonically increasing mean recall curve. Both, the precision and recall are quite low with the maximum mean recall being around 0.3 and the maximum mean precision being 0.58.
- Since, both precision and recall are low, the F-measure also tends to be low, peaking at a rank of 5 or 6 depending upon the segmenter and tokenizer.
- Similar to the F-score, nDCG also appears to first increase with increasing rank, then peaks at around a rank of 3 following which it decreases with increasing rank. This tells us that if the top-3 results are taken, their ranking is closest to the ideal ranking for those 3 retrieved documents.
- In comparison to mean precision and mean recall, MAP and nDCG are seen to have higher values. This is because, in case of MAP and nDCG, at every rank, not only the number of relevant documents are considered, but the ranks of these relevant documents are also considered. At larger Ks we expect the mean precision to drop drastically even for well-performing IR systems since the relevant documents would have already been retrieved at the top ranks, while the bottom ranks would mostly have irrelevant documents. Mean precision at these large ranks would not bother about how the system performed at the top ranks, leading to a low value, giving the impression that the system performs badly.
- Across the different kinds of segmenters and tokenizers, we notice that (Punkt, PennTreeBank) performs the best followed by (Naive, Naive), while (Punkt, Naive) follows closely. However, (Naive, PennTreeBank) leads to a drastic deterioration in performance. This may be because of the assumptions that have gone into formulating the Naive segmenter and the way it is handled these segmented sentences are handled by the Penn Tree Bank tokenizer.
- The maximum nDCG ≈ 0.6317 is obtained upon using the Punkt segmenter and PennTreeBank Tokenizer, at a rank of 3.

k	mean Precision	mean Recall	mean F-Score	MAP	mean nDCG
1	0.5733333333333334	0.09589924824630701	0.1579381737521383	0.5733333333333334	0.5733333333333334
2	0.45555555555555555	0.14587211620423243	0.2077839520386236	0.6044444444444445	0.5922178145992304
3	0.39111111111111085	0.18048582557812842	0.22988805415612254	0.6170370370370369	0.6078623676137987
4	0.3377777777777778	0.20443456334525703	0.23596656234168703	0.6180246913580247	0.6128377625663775
5	0.29955555555555585	0.22224155011302807	0.2355191126842941	0.6095061728395064	0.6090237447230771
6	0.27555555555555544	0.24166786481913707	0.23695214540520237	0.6031407407407413	0.6145033386765281
7	0.2546031746031749	0.25684339916133814	0.2356206075359057	0.5949922398589068	0.6139670593779312
8	0.2388888888888889	0.2699540866708087	0.23344638481106544	0.5835148148148152	0.6061931229209179
9	0.22271604938271625	0.2830014343848232	0.22965617946718606	0.5766165784832453	0.6012342604605938
10	0.20977777777777792	0.2943964739982536	0.2258550875429073	0.5676497522465778	0.5961502790565588

Table 1: -Segmenter Punkt ,-Tokenizer Naive

k	mean Precision	mean Recall	mean F-Score	MAP	mean nDCG
1	0.5822222222222222	0.09861901736607613	0.16237144418540872	0.5822222222222222	0.5822222222222222
2	0.47333333333333333	0.15376760239445553	0.2186571647615095	0.6244444444444445	0.611475122578278
3	0.4148148148148144	0.19121462270692557	0.24364373420592025	0.6362962962962962	0.6261400480430992
4	0.3566666666666667	0.21403482111218153	0.2477834281193371	0.6375308641975308	0.6317551017595764
5	0.32266666666666705	0.23668836028790852	0.25222896354894375	0.6290987654320987	0.6278744599664791
6	0.2933333333333333	0.25566067192688685	0.25210598285315744	0.6227950617283952	0.6296963394104249
7	0.26984126984127005	0.2705175706688431	0.24912323776274103	0.6154292768959436	0.6265422268947467
8	0.25	0.2815790678624567	0.24427124258273775	0.6074648526077098	0.6227500110606906
9	0.2330864197530867	0.2937785227285782	0.23983156165846176	0.6020014424288234	0.6199465815517267
10	0.2195555555555557	0.3057563437581232	0.2359331556271147	0.5956796772206824	0.617710915202499

Table 2: -Segmenter Punkt, -Tokenizer Penn Treebank

k	mean Precision	mean Recall	mean F-Score	MAP	mean nDCG
1	0.5822222222222222	0.0968019751490339	0.15955988537384994	0.5822222222222222	0.5822222222222222
2	0.46	0.14848587281798906	0.21115784974585466	0.6133333333333333	0.6012276508710707
3	0.4074074074074071	0.18707575027331627	0.23874503667977173	0.6222222222222222	0.6109604439864964
4	0.3566666666666667	0.21154112221848262	0.2459249414871057	0.6209876543209877	0.6160423479501275
5	0.31377777777777816	0.22972088435902907	0.2447859671843787	0.6160185185185186	0.6149078243151739
6	0.28814814814814793	0.2502655560168284	0.2467076313581968	0.6140320987654322	0.62300427421539
7	0.2660317460317463	0.2663780484626542	0.24528817640264833	0.6061169312169311	0.6215274844951625
8	0.2461111111111111	0.2786070941571496	0.24072018923292282	0.5969945326278661	0.6168763411856127
9	0.22913580246913612	0.29017296038968254	0.2361476891454576	0.5912767573696146	0.61475882814221
10	0.21422222222222237	0.2995140477824939	0.2304332337090298	0.5830518476526413	0.609540038505747

Table 3: -Segmenter naive, -Tokenizer naive

k	mean Precision	mean Recall	mean F-Score	MAP	mean nDCG
1	0.4222222222222222	0.06959801244507122	0.11516578637190777	0.4222222222222222	0.4222222222222222
2	0.34	0.10884720674072647	0.15562999488466664	0.45111111111111113	0.43939074059249655
3	0.29629629629629634	0.1350229921152948	0.17309849492259102	0.4562962962962962	0.4484209797540474
4	0.2588888888888889	0.15482681800417833	0.17979353876746781	0.4516049382716049	0.4458860680808825
5	0.23200000000000007	0.1708888111079671	0.18210573208102696	0.4511419753086421	0.4534409773687639
6	0.20888888888888885	0.18123112311694575	0.1791900895413774	0.4495753086419755	0.45412539638600014
7	0.19365079365079377	0.19177447121201802	0.17808766625139885	0.44296208112874785	0.45076775361262966
8	0.17722222222222223	0.20023564839158603	0.17360310142441074	0.439286722096246	0.4497383537612789
9	0.16246913580246916	0.20736086885013988	0.16848173357479304	0.4400413076341648	0.4535070828568443
10	0.15155555555555586	0.21432735914996356	0.1647070961388866	0.4369567271352986	0.4538978197860039

Table 4: -Segmenter naive, -Tokenizer Penn Treebank

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.

Answer:

Yes, for 41 out of the 225 queries, the IR system is unable to retrieve a single relevant document with all the scores tending to zero. Table 5 lists them all. Inspecting these queries shows that these queries are highly specific and granular which makes it difficult to retrieve relevant documents. Also, another observation is that some of the queries had very few ground truth relevant documents which made the retrieval task further difficult. For example, the query (query ID = 22), ’*did anyone else discover that the....*’ is a vague query and also its ground truth relevant documents are only 2 which makes it harder for it to retrieve the relevant documents, thereby resulting in a low evaluation score compared to the some of the generalized queries where the retrieval system has performed well (Examples given in Table 6). Similarly we were not able to retrieve relevant documents for the query (queryID = 116), ’*what is the magnitude and distribution.....*’ which is highly specific in nature.

9	76	119	181
12	78	133	204
19	79	141	205
22	85	143	207
28	87	150	216
44	88	151	217
61	95	152	218
62	98	153	219
63	110	154	
66	115	167	
74	116	180	

Table 5: Query IDs without a single relevant document retrieved

Query ID	Precision	Recall	F-Score	Average Precision	nDCG	Query
3	0.8	0.888889	0.842105	0.986111	0.874226	what problems of heat conduction in composite slabs have been solved so far
185	0.8	0.8	0.8	0.908532	0.894438	experimental studies on panel flutter
101	0.6	0.857143	0.705882	0.855556	0.913175	why does the incremental theory and the deformation theory of plastic stress-strain. . . ..
120	0.7	0.7	0.7	0.885714	0.80797	are previous analyses of circumferential thermal buckling of. . . .
92	0.8	0.571429	0.666667	1	0.99222	given complete freedom in the design of an airplane, what procedure would be used in order to minimize sonic boom. . .

Table 6: Good IR performance for the above Query_IDs

Note : The above queries were identified on thorough inspection of the performance of the IR system over all the queries.[All the results in Reference-1]

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

- The model is unable to account for the sequence in which the words in the document or query occurs. It treats them like a bag of words.
- The model is unable to handle synonymy. For example, if there is a collection of documents containing the word love but none containing the word affection, but the query contains the word affection, even though the documents containing love are the most apt, they will not be retrieved since they do not contain the word affection exactly. Thus, the model cannot associate words with their meanings since each word is considered orthogonal to the other.
- The model is unable to differentiate between polysemous entries. For example, if the query is "Ganga river bank", documents on "World Bank" are also retrieved since they both contain the word bank, even though they are unrelated. This happens because the model solely looks at the presence of the word bank in both and decides that they must be similar, it cannot differentiate between their different meanings and looks at the forms only.
- Each time a new type is introduced, either in the query or in the document, all the vectors need to be reinitialzed and calculated, since it leads to the addition of a new dimension

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?

Answer:
To weigh the contribution of the title, we can consider the document as body + title. Then, we could reformulate the TF-IDF of the document as:

$$\text{TF-IDF}(\textit{document}) = (\text{TF-IDF}(\textit{title}) * \alpha) + (\text{TF-IDF}(\textit{body}) * (1 - \alpha))$$

Thus, if the word is present only in title, then the weights of body for that particular word will not add to the TF of that word, and vice versa. To weigh the title thrice as much as the body, we can take $\alpha = \frac{3}{4} = 0.75$

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

Answer:
The advantages are that the precision increases, since the sequence of words is now considered. For example, if the query has King Kong, Hong Kong will not be retrieved, this reducing false positives. The disadvantages include decrease in recall. Further, dimensions of the

space increase greatly since each possible bigram from our vocabulary forms a dimension in the vector space model.

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.

Answer:
In order to record feedback from the user we can analyse their response to the result. If the titles are important, we can say that for a particular query, the links which have larger number of clicks are more relevant. However, if the titles are not discriminative then we can record the amount of time spent on a page, with the assumption that if a result is relevant, more time is spent by the user in reading it, while if a result is irrelevant, a quick glance will reveal so to the user. Further we could also take note of the number of downloads in a page.

References

1. IR system Evaluation on all the queries
2. TF-IDF representation of a query and document
3. Similarity measures for TF-IDF vectors