CS6370 : Natural Language Processing — Assignment #2

Roll No: CS14B017 — Full Name: Manohar Mulle

Roll No: CS14B043 — Full Name: Harshal Gawai

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 table of term frequencies for three documents. Give an inverted index representation for the same.

	Doc 1	**Doc 2**	**Doc 3**
cat	4	2	1
dog	3	0	3
animal	1	3	3

Solution: Given term frequency matrix:
$\Longrightarrow$

	Doc 1	**Doc 2**	**Doc 3**
cat	4	2	1
dog	3	0	3
animal	1	3	3

Sort terms alphabetically
$\Longrightarrow$

	Doc 1	**Doc 2**	**Doc 3**
animal	1	3	3
cat	4	2	1
dog	3	0	3

Now, Sort documents (will remain same)
$\Longrightarrow$

	Doc 1	**Doc 2**	**Doc 3**
animal	1	3	3
cat	4	2	1
dog	3	0	3

Getting doc frequencies and posting lists for each term to get inverted index represents
$\Longrightarrow$

Term	Doc. freq.	Posting list
animal	3	$\rightarrow \boxed{A} \rightarrow \boxed{B} \rightarrow \boxed{C}$
cat	3	$\rightarrow \boxed{A} \rightarrow \boxed{B} \rightarrow \boxed{C}$
dog	2	$\rightarrow \boxed{A} \rightarrow \boxed{C}$

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.

Solution: Given term frequency matrix:
$\Longrightarrow$

	Doc 1	**Doc 2**	**Doc 3**
cat	4	2	1
dog	3	0	3
animal	1	3	3

Let TF-IDF vector representations for the documents in the above table be

$$\overline{D}_j = \sum_{t \in \mathbb{T}} w_{t,d} \hat{T}_t \tag{1}$$

Where $d \in \mathbb{D} = \{A, B, C\}$
$t \in \mathbb{T} = \{cat, dog, animal\}$
$\hat{T}_t$ = unit vector corresponding to term $t \in \mathbb{T}$ that is orthogonal to $\hat{T}_t$, $\forall t' \in \mathbb{T} \backslash \{j\}$
$w_{t,d}$ = TF-IDF measure weight of (t,d) term-document pair

Now, we will find $w_{t,d}$ $\qquad \forall d \in \mathbb{D}$ and $\forall t \in \mathbb{T}$

$$\begin{aligned} w_{t,d} &= tf\text{-}idf_{t,d} \\ &= tf_{t,d} * idf_t \end{aligned}$$

$$idf_t = log \frac{N}{df_t}$$

here, $tf_{t,d}$ = term frequency of term t in document d
and idf_t = inverse document frequency of term t
N = total no. of documents in which term t appears
$N = |\mathbf{D}| = 3$

Count (t, d) is already given:

	A	B	C
cat	4	2	1
dog	3	0	3
animal	1	3	3

Calculating term frequency $tf_{t,d}$:

	A	B	C	df	$idf_t = log\frac{N}{t_f}$
cat	$\frac{4}{8}$	$\frac{2}{5}$	$\frac{1}{7}$	3	$log\frac{3}{3} = 0$
dog	$\frac{3}{8}$	$\frac{0}{5}$	$\frac{3}{7}$	2	$log\frac{3}{2} = log1.5 = 0.176$
animal	$\frac{1}{8}$	$\frac{3}{5}$	$\frac{3}{7}$	3	$log\frac{3}{3} = 0$

Calculating tf-$idf_{t,d}$ measure:

	A	B	C
cat	0	0	0
dog	$\frac{3}{8} * log1.5 = 0.066$	0	$\frac{3}{7} * log1.5 = 0.076$
animal	0	0	0

	cat	dog	animal
A	0	$\frac{3}{8} * log1.5 = 0.066$	0
B	0	0	0
C	0	$\frac{3}{8} * log1.5 = 0.076$	0

$\therefore$ the tf-$idf_{t,d}$ vector representation of the documents is

$$\overline{D}_A = 0 \cdot \hat{T}_{cat} + 0.066 \cdot \hat{T}_{dog} + 0 \cdot \hat{T}_{animal}$$
$$\overline{D}_B = 0 \cdot \hat{T}_{cat} + 0 \cdot \hat{T}_{dog} + 0 \cdot \hat{T}_{animal}$$
$$\overline{D}_C = 0 \cdot \hat{T}_{cat} + 0.076 \cdot \hat{T}_{dog} + 0 \cdot \hat{T}_{animal}$$

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

Solution:

Term	Doc. freq.		Posting list
animal	3	$\rightarrow$	$\boxed{A} \rightarrow \boxed{B} \rightarrow \boxed{C}$
cat	3	$\rightarrow$	$\boxed{A} \rightarrow \boxed{B} \rightarrow \boxed{C}$
dog	2	$\rightarrow$	$\boxed{A} \rightarrow \boxed{C}$

Based on this inverted index that was constructed in que.1 **Doc A** and **Doc C** would be retrieved.

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

Solution:
The query vector here is

$$\begin{aligned}\overline{D}_q &= \sum_{t\in\mathbb{T}} w_{t,d}\hat{T}_t \\ &= \sum t \in \mathbb{T} tf_{t,q} \times idf_t \overline{T}_t\end{aligned}$$

from quesiton 2, for $t = "dog, idf_t = 0.176$

The given query doc q is a single word doc containing the word "dog"
$\implies$ the term frequency w.r.t. q is $tf_{t,q} = 1$ for t="dog"
$= 0$ otherwise

$$\implies \overline{D}_q = 0.176\hat{T}_d og \tag{2}$$

Now, cosine similarity between $\overline{D}_q$ and $\overline{D}_d \forall d \in \mathbb{D}$ is $\cos\theta_{(q,d)}$
Now, cosine similarity between $\overline{T}_q$ and $\overline{T}_d$ $\forall \in \{A, B\}$
Cosine similarity between $\overline{T}_q$ and $\overline{T}_d$

$$\begin{aligned}sim(\overline{T}_q, \overline{T}_A) &= \cos\theta_{q,A} \\ &= \frac{\overline{T}_q.\overline{T}_A}{|\overline{T}_q||\overline{T}_A|} \\ &= 1\end{aligned}$$

Cosine similarity between $\overline{T}_q$ and $\overline{T}_B$:

$$\begin{aligned} sim(\overline{T}_q, \overline{T}_B) &= \cos\theta_{q,B} \\ &= \frac{\overline{T}_q . \overline{T}_B}{|\overline{T}_q||\overline{T}_B|} \\ &= 1 \end{aligned}$$

5. 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.

Solution: (Implemented in code)

6. (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?

Solution:
Ans. (a)
IDF i.e. inverse Document frequency of term t is given by:

$$idf_t = log\frac{N}{df_t} \tag{3}$$

here, df_t = document frequency of a term t
$N =$ total no. of documents in which term t appears
So, if a term occures in every document $\implies df_t = N$

$$\begin{aligned} idf_t &= log\frac{N}{df_t} \\ &= log\frac{N}{N} \\ &= 0 \end{aligned}$$

$\therefore$ IDF is 0 if a term occures in every document

Ans. (b)
If certain word doesn't appear in any of the documents
$\implies df_t$ (document frequency) is 0

In this case $log\frac{N}{0}$ won't be finite
Hence, IDF of such term will NOT be finite

A light modification to above IDF formula so that denominator won't be 0:

$$idf_t = log\frac{N+1}{df_t+1} \tag{4}$$

$$\tag{5}$$

Here, df_t for a word that won't appear in any document will be finite.
(**Note:** idf_t will be high but finite.)

7. 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.

Solution: One can use the Euclidean distance as a measure that can be used to compare vectors.
Euclidean distance (squared) between x,y:

$$(||x-y||)^2 = (||x||)^2 + (||y||)^2 - 2 < x, y > \tag{6}$$

where, x and y are two vectors or two documents on the Vector Space
If one don't normalize the vectors to be all the same length then their length will depend on the length of the document.
In document classification we don't want to be biased by the document lengths.
This is one reason why cosine similarity is preferred choice over Euclidean distance.

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

Solution: Accuracy is the simplest model evaluation metric for classification models. It is the percentage of correctly predicted labels.

$$Accuracy = \frac{Numberofcorrectpredictions}{Totalnumberofpredictions} \tag{7}$$

We tend to use accuracy because everyone has an idea of what it means rather than because it is the best tool for the task.When using accuracy, we are assign equal cost to false positives and false negatives.
Precision-Recall is a useful measure of success of prediction when the classes are very imbalanced.
In information retrieval, precision is a measure of result relevancy, while recall is a

measure of how many truly relevant results are returned.
There are many other metrics for evaluating binary classification systems. Even plotting a graph can be helpful.
There is no single best way to evaluate any system, but different metrics gives different (and valuable) insights into how any classification model performs.

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

Solution: The $F\text{-}measure$ is calculated as the harmonic mean of **precision** and **recall**, giving each the same weightage. $F\text{-}measure$ is given by:

$$F\text{-}measure = \frac{2 * precision * recall}{precision + recall} \tag{8}$$

$F_\alpha\text{-}measure$ is given by:

$$F_\alpha\text{-}measure = \frac{1}{\dfrac{\alpha}{precision} + \dfrac{1-\alpha}{recall}} \tag{9}$$

clearly, for $\alpha < 0.5, F_\alpha\text{-}measure$ gives more weightage to recall than for precision

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

Solution: A limitation of precision@k is that it doesn't consider the position of the relevant items.
Average Precision is a metric that evaluates whether all of the relevant items selected by the model are ranked higher or not.

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

Solution:
1. The average precision $AP(q)@K$ for a query q is essentially the precision $@k$ for q with the positions of relevant-retrieved documents taken into account.
2. The same set of retrieved documents gives different $AP(q)@k$ based on different positions of the relevant docs among retrieved documents.
3.The more the relevant docs appear on top of the retrieved list, the higher $AP(q)@k$

Mean average precision for a set of queries is the mean of the average precision scores over all queries.

$$MAP = \frac{\sum_{q=1}^{Q} AP(q)}{Q} \tag{10}$$

where,
Q is the total number of queries.
$AP(q)$ is the average precision for query q.

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

Solution: The DCG accumulated at a particular rank position p is defined as:

$$DCG_p = \sum_{i=1}^{p} \frac{rel_i}{log_2(i+1)} \tag{11}$$

$$nDCG = \frac{DCG_p}{IDCG_p} \tag{12}$$

Average Precision assumes binary relevance. Which means an item is either interest or not (i.e. value is 0 or 1). Whereas NDCG allows relevance scores to be any positive real number.
Since Cranfield dataset contains non-binary relevance judgements for queries, nDCG is a better metric than AP as it can make use of more information(relevance values).

13. (Implemented in Code)
14. (Implemented in Code)
15. 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.
16. Do you find any shortcoming(s) in using a Vector Space Model for IR? If yes, report them.

Solution: Vector Space Model for IR suffers from two major shortcomings:
1. It makes the consideration of all words impractical: since each word is a dimension, considering all words would imply expensive computations in a very high-dimensional space.
2. It assumes that all words are independent.
3. It does not take into consideration neither the semantics nor the context of the of the sentences of the documents.

17. 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?

Solution: Let $\overline{V}_{old}$ be the vector representation of a Doc d just considering terms from its text.
Let $\overline{V}_{d_{title}}$ be the vector representation of title of d
Now, one way to include the title while representing d as a vector is,

$$\overline{V}_{d_{new}} = \alpha(\overline{V}_{d_{old}}) + (1-\alpha).\overline{V}_{d_{title}}, \qquad 0 \leq \alpha \leq 1 \tag{13}$$

Giving title 3 times the weight of the document text:

$$\begin{aligned} \implies \qquad & \frac{1-\alpha}{\alpha} = 3 \\ \implies \qquad & \alpha = 0.25 \\ \implies \qquad & \overline{V}_{d_{new}} = 0.25.\overline{V}_{d_{old}} + 0.75.\overline{V}_{d_{title}} \end{aligned}$$

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

Solution: Advantages:
Hits will be more powerful
Helps in retreiving only VERY relevant documents.

Disadvantages:
The size of the bigram vocabulary is likely to be very high(worst-case:$O(n^2)$) compared to that of a unigram
There will be less hits. Many times just because 2 words doesn't appear side by side, it doesn't mean that the document is not relevant.

19. 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

Solution: One straight-forward way to get relevance feedback is based on the number of clicks a document get as a whole from all the users for a query.

This is a non-intrusive way where we don't have to ask users in specific and the entire process(keeping track of user clicks) can be dome in the background.

Relevance can be updated(increased/decreased) in a timely manner based on the number of clicks a document gets for a set of "similar" queries in a given period of time