Introductory topics and similarity models

Assignment 1

Note: Use log with base 2 for all computations

Task 1: Basic definitions

Information retrieval vs Data retrieval
Data retrieval, is in the context of an IR system a method for determining which documents of a collection which contains the keywords in a query given. It is about searching for a list of keywords, and returning the instances containing them. Information retrieval, on the other hand, is more about finding the relevant information the searcher requests. This includes documents which includes synonyms of the keywords, even without the keywords present, and documents including the keyword even though the keyword may have been misspelled. Thus, you can define IR as the operation of finding relevant *information* in a collection of documents, neglecting inaccuracy in the keywords or bad spelling.

Structured vs Unstructured data
One may define the difference of structured or unstructured data in many ways. First of all, it is about whether the data is highly organized and factual, or *quantitative,* versus don't having any predefined structure and coming in with all it's diversity, often described as *qualitative. Structured data rely on having some predefined formats, where unstructured data can come in all different formats and sizes. Lastly, structured data usually comes in text only format, where unstructured data can be either text, images, sound, video etc.*

Task 2: Term weighting

Term frequency (tf) **-** $[0, \infty) \in Z$
Term frequency is simply how often a term occurs in a document. In a natural language context, terms corresponds to words or phrases, but could also represent any token in text.

Document frequency (df) **-** $[0, \infty) \in Z$
Document frequency is on the other side the number of documents containing a particular term. This measure is often used eliminate unimportant words from analysis. For example, if you have words that appeared in at least 80% of the documents, it can be removed as this is likely to be a non-special word in an analysis point of view.

Inverse document frequency (idf) **-** $[0, \infty) \in R$
Inverse document frequency is very simply the inverse of document frequency, as given by the name. If you take the total number of documents, divided by the number of documents containing the term (df), you will get the inverse document frequency. This tells us how rare a term is - if many documents contain the term, the divisor will be large, resulting in a low value.

Why idf **is important for term weighting**
Document frequency measures commonness, and we prefer to measure rareness, as this is what is giving the term importance in the documents we are searching in. The idf measure gives us this rareness.

> We also have *tf-idf,* which is a numeric measure of how important a term is within a document.

Task 3: IR Models

The following two subtasks uses this dataset:

$doc1 = \{Big, Cat, Small, Dog\}$
$doc2 = \{Dog\}$
$doc3 = \{Cat, Dog\}$
$doc4 = \{Big, Cat, Big, Small, Cat, Dog\}$
$doc5 = \{Big, Small\}$
$doc6 = \{Small, Cat, Dog, Big\}$
$doc7 = \{Big, Big, Big\}$
$doc8 = \{Dog, Cat, Cat\}$
$doc9 = \{Cat, Small\}$
$doc10 = \{Small, Small, Big, Dog\}$

Subtask 3.1: Boolean model and vector space model

q1 = "Cat AND Dog"

q2 = "Cat AND Small"

q3 = "Dog OR Big"

q4 = "Dog NOT Small"

q5 = "Cat"

1 **Which of the documents will be returned as the result for the above queries using the Boolean model? Explain your answers and draw a figure to illustrate**

The first one will include all documents both containing "Cat" and "Dog", giving the result

$$q1 = \{doc1, doc3, doc4, doc6\}$$

The second one will include all documents both containing "Cat" and "Small", giving the result

$$q2 = \{doc1, doc4, doc6, doc9\}$$

The third one will include all documents containing either "Dog" or "Big", giving the result

$$q3 = \{doc1, doc2, doc3, doc4, doc5, doc6, doc7, doc8, doc10\}$$

The fourth one will include all the documents which contains the word "Dog" and not the word "Small", giving the result

$$q4 = \{doc2, doc3, doc8\}$$

The last one will include all the documents containing the term "Cat", giving the result

$$q5 = \{doc1, doc3, doc4, doc6, doc8, doc9\}$$

2 What is the dimension of the vector space representing this document collection when you use the vector model and how is it obtained?

Each dimension represents a term. Our vocabulary in this task consists of four words, being "Cat", "Dog", "Small" and "Big". It is obtained by finding how many unique values there is in the set. The dimension is therefore the size of the vocabulary, being 4.

3 Calculate the weights for the documents and the terms using tf and idf weighting. Put these values into a document-term-matrix.

For tf, we use the formula $1 + log_2(frequency)$ if tf is larger than 0, where frequency is the number of occurrences. If the terms do not appear in the document, the value is 0.

	“Big”	**“Small”**	**“Cat”**	**“Dog”**
doc1	1	1	1	1
doc2	0	0	0	1
doc3	0	0	1	1
doc4	2	1	2	1
doc5	1	1	0	0
doc6	1	1	1	1
doc7	2,58	0	0	0
doc8	0	0	2	1
doc9	0	1	1	0
doc10	1	2	0	1

For idf, we use $log_2(N/df)$, where N is the total number of documents and df is the amount of documents containing the word.

idf	**“Big”**	**“Small”**	**“Cat”**	**“Dog”**
df	6	6	6	7
IDF	0,74	0,74	0,74	0,51

Lastly, for creating the matrix, we use $tf - idf$ weighting by multiplying the tf and idf

tf x idf	“Big”	“Small”	“Cat”	“Dog”
doc1	0,74	0,74	0,74	0,51
doc2	0	0	0	0,51
doc3	0	0	0,74	0,51
doc4	1,47	0,74	1,47	0,51
doc5	0,74	0,74	0	0
doc6	0,74	0,74	0,74	0,51
doc7	1,91	0	0	0
doc8	0	0	1,47	0,51
doc9	0	0,74	0,74	0
doc10	0,74	1,47	0	0,51

4 Study the documents 2, 3, 5 and 7 and compare them to document 9. Calculate the similarity between document 9 and these four documents according to Euclidean distance.

For this calculation, we use the Euclidean distance formula:

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + (p_3 - q_3)^2 + (p_4 - q_4)^2}$$

Giving the following similarities:

$d(doc9, doc2) = \sqrt{(0-0)^2 + (0.74-0)^2 + (0.74-0)^2 + (0-0.51)^2} = 1.162$
$d(doc9, doc3) = \sqrt{(0-0)^2 + (0.74-0)^2 + (0.74-0.74)^2 + (0-0.51)^2} = 0.899$
$d(doc9, doc5) = \sqrt{(0-0.74)^2 + (0.74-0.74)^2 + (0.74-0)^2 + (0-0)^2} = 1.042$
$d(doc9, doc7) = \sqrt{(0-1.91)^2 + (0.74-0)^2 + (0.74-0)^2 + (0-0)^2} = 2.171$

5 Rank the documents for query q5 using cosine similarity.

The function for cosine similarity:

$$sim(d_1, d_2) = \frac{\overrightarrow{V}(d_1) \cdot \overrightarrow{V}(d_2)}{|\overrightarrow{V}(d_1)| \cdot |\overrightarrow{V}(d_2)|}$$

$$sim(doc1, q5) = \frac{(x_1 * y_1) + (x_2 * y_2) + (x_3 * y_3) + (x_4 * y_4)}{\sqrt{x_1^2 + x_2^2 + x_3^2 + x_4^2} * \sqrt{y_1^2 + y_2^2 + y_3^2 + y_4^2}}$$

For document 1 and query 5, the cosine similarity will be:

$sim(doc1, q5) = \frac{(1*0)+(1*0)+(1*1)+(1*0)}{\sqrt{1^2+1^2+1^2+1^2}*\sqrt{0^2+0^2+1^2+0^2}} = 0.50$
$sim(doc2, q5) = \frac{(0*0)+(0*0)+(0*1)+(1*0)}{\sqrt{0^2+0^2+0^2+1^2}*\sqrt{0^2+0^2+1^2+0^2}} = 0.00$
$sim(doc3, q5) = \frac{(0*0)+(0*0)+(1*1)+(1*0)}{\sqrt{0^2+0^2+1^2+1^2}*\sqrt{0^2+0^2+1^2+0^2}} = 0.71$
$sim(doc4, q5) = \frac{(2*0)+(1*0)+(2*1)+(1*0)}{\sqrt{2^2+1^2+2^2+1^2}*\sqrt{0^2+0^2+1^2+0^2}} = 0.63$
$sim(doc5, q5) = \frac{(1*0)+(1*0)+(0*1)+(0*0)}{\sqrt{1^2+1^2+0^2+0^2}*\sqrt{0^2+0^2+1^2+0^2}} = 0.00$
$sim(doc6, q5) = \frac{(1*0)+(1*0)+(1*1)+(1*0)}{\sqrt{1^2+1^2+1^2+1^2}*\sqrt{0^2+0^2+1^2+0^2}} = 0.50$
$sim(doc7, q5) = \frac{(3*0)+(0*0)+(0*1)+(0*0)}{\sqrt{3^2+0^2+0^2+0^2}*\sqrt{0^2+0^2+1^2+0^2}} = 0.00$
$sim(doc8, q5) = \frac{(0*0)+(0*0)+(2*1)+(1*0)}{\sqrt{0^2+0^2+2^2+1^2}*\sqrt{0^2+0^2+1^2+0^2}} = 0.89$
$sim(doc9, q5) = \frac{(0*0)+(1*0)+(1*1)+(0*0)}{\sqrt{0^2+1^2+1^2+0^2}*\sqrt{0^2+0^2+1^2+0^2}} = 0.71$
$sim(doc10, q5) = \frac{(1*0)+(2*0)+(0*1)+(1*0)}{\sqrt{1^2+2^2+0^2+1^2}*\sqrt{0^2+0^2+1^2+0^2}} = 0.00$

Giving the ranking:

1. doc8
2. doc3 doc9
3. doc4
4. doc1 doc6
5. doc2 doc5 doc7 doc10

Or in a line order:

doc8, [doc3 doc9], doc4, [doc1 doc6], [doc2 doc5 doc7 doc10]

Subtask 3.2: Probabilistic models

$$q_1 = \{Cat, Dog\}$$
$$q_2 = \{Small\}$$

1 What are the main differences between BM25 model and the probabilistic model introduced by Robertson-Jones?

As I have interpreted the subject matter, the model introduced by Robertson-Jones was mainly intended as a framework for further models. This model misses some key points for working as an actual algorithm for weighting documents. The model is not accurate for estimating the first round of probabilities, it lacks index terms not weighted, and terms is assumed mutually independent. BM25 on the other hand, is a collection of scoring-functions weighting documents, with different shapes and parameters.

2 Rank the documents using the BM25 model.

Set the parameters to $k = 1.2$ *and* $b = 0.75$. *(We can assume that relevance information is not provided). A hint is also given to use* $idf = log(\frac{N}{df_t})$ *in the BM25 model for avoiding negative numbers.*

$$\sum_{i}^{n} IDF\left(q_1\right) \frac{tf \cdot (k+1)}{tf + k \cdot \left(1 \cdot b + b \cdot \frac{fieldLen}{avgFieldLen}\right)}$$

Where $IDf = \log_2\left(\frac{N}{df_t}\right)$, $k = 1.2$, $b = 0.75$ and $avgFieldLen = 3.16$

Rewrites and gets the following formula:

$$BM25 = \sum_{t \in q} \log_2\left(\frac{N}{dt_t}\right) \cdot \frac{(k+1)\,tf}{k \cdot \left((1 \cdot b) + b \cdot \frac{fieldLen}{avgFieldLen}\right) + tf}$$

Lets show the process of query 1 and 2 on doc1:

$$q_{1_{BM25}} = \log_2\left(\frac{10}{6}\right) \cdot \frac{(1.2+1)}{1.2 \cdot \left((1-0.75) + 0.75 \cdot \frac{4}{3.1}\right)} + \log_2\left(\frac{10}{7}\right) \cdot \frac{(1.2+1)}{1.2 \cdot \left((1-0.75) + 0.75 \cdot \frac{4}{3.1}\right)} = 1.884$$

$$q_{2_{BM25}} = \log_2\left(\frac{10}{7}\right) \cdot \frac{(1.2+1)}{1.2 \cdot \left((1-0.75) + 0.75 \cdot \frac{4}{3.1}\right)} = 1.110$$

This was done for all the remaining documents. One thing to notice is that when tf is 0, the BM25 will also become 0 for that query, which makes sense as the numerator in the fraction is a product of the term frequency. I calculated all the next formulas using a spreadsheet, and got the following results:

	q1	q2
doc1	1,884	1,110
doc2	1,918	0
doc3	3,127	0
doc4	1,348	0,794
doc5	0	1,841
doc6	1,884	1,110
doc7	0	0
doc8	2,351	0
doc9	1,841	1,841
doc10	0,775	1,110

Which gives the following ranking:

	q1			q2
doc3	3,127		doc5	1,841
doc8	2,351		doc9	1,841
doc2	1,918		doc1	1,110
doc1	1,884		doc6	1,110
doc6	1,884		doc10	1,110
doc9	1,841		doc4	0,794
doc4	1,348		doc2	0
doc10	0,775		doc3	0
doc5	0		doc7	0
doc7	0		doc8	0

Or in a line order:

q1 = doc3, doc8, doc2, [doc1 doc6], doc9, doc4, doc10, [doc5 doc7]

q1 = [doc5 doc9], [doc1 doc6 doc10], doc4, [doc2 doc3 doc7 doc8]