*Magnus Nermark, Kristian Skog*
*Paul Mitchell, Fredrik Gusland*

## Task 1: Basic Definitions

*Explain the main differences between:*

1. **Information Retrieval vs Data Retrieval**
   An information retrieval system manages unstructured data, while a data retrieval system manages structured data with well-defined semantics. In addition, an information retrieval system allows partial matching of a single query with multiple ranked results, whereas a data retrieval system operates with exact results. Information retrieval systems also handle content characterization by representing documents by index terms, which are words in a document that characterizes content by analysing its semantics.

2. **Structured Data vs Unstructured Data**
   Unstructured data refers to information that either is not organized in a predefined manner or does not have a predefined data model. Retrieval of unstructured data is based on the semantics of its content. Examples of unstructured data include text files such as PDFs, Word files, and email content. Structured data refers to information that is highly organized such that it can easily be included in a relational database - it is usually structured in rows and columns such that a query produces an exact result. Examples of structured data includes XML and JSON.

## Task 2: Term Weighting

*Explain:*

1. **Term Frequency (tf):**
   Term frequency, tf, is simply the number of times a term occurs in a certain document.

2. **Document Frequency (df):**
   Document frequency, df, is the number of documents in a collection of documents which contain a certain term.

3. **Inverse Document Frequency (idf):**
   Inverse document frequency, idf, is a formula to calculate the commonality of a certain term in a collection of documents (corpus). The formula assigns weights to terms based on the number of documents in a corpus in which a certain term occurs.

4. **Why idf is important for term weighting:**
   Inverse document frequency increases the weight of terms that occur rarely in a document collection - this is important, because the word “the” is likely to occur multiple times in a single document, but does not carry much meaning, whereas the word “nuclear” may be much less common, but carries a lot more meaning to a document.

## Task 3: IR Models

*Given the following document collection containing words from the set O = {Apple, Samsung, HTC}, answer the questions in subtasks 3.1 and 3.2.*

*doc1 = {Apple HTC HTC}*
*doc2 = {Apple Samsung}*
*doc3 = {Samsung Samsung HTC}*
*doc4 = {Apple Samsung}*
*doc5 = {Samsung}*
*doc6 = {Apple Samsung Samsung HTC Samsung HTC HTC HTC}*
*doc7 = {HTC Apple}*
*doc8 = {Samsung Samsung Samsung}*
*doc9 = {HTC Apple HTC}*
*doc10 = {Apple Apple Apple Samsung HTC}*

### *SubTask 3.1: Boolean Model and Vector Space Model*

*Given the following queries:*

*q1 = "Apple AND Samsung"*
*q2 = "Apple AND Microsoft"*
*q3 = "Apple OR HTC"*
*q4 = "Samsung NOT HTC"*
*q5 = "Apple"*

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.

| K = {k1 = Apple, k2 = Samsung, k3 = HTC} | |
|---|---|
| c(doc1) = (1, 0, 1) | c(doc6) = (1, 1, 1) |
| c(doc2) = (1, 1, 0) | c(doc7) = (1, 0, 1) |
| c(doc3) = (0, 1, 1) | c(doc8) = (0, 1, 0) |
| c(doc4) = (1, 1, 0) | c(doc9) = (1, 0, 1) |
| c(doc5) = (0, 1, 0) | c(doc10) = (1, 1, 1) |

**q1 = "Apple AND Samsung"**

q1DNF = (1, 1, 1) v (1, 1, 0)
sim(dj, q1) = (0, 1, 0, 1, 0, 1, 0, 0, 0, 1)
**{doc2, doc4, doc6, doc10}**
These are the only documents that contain both the words Apple and Samsung.

**q2 = "Apple AND Microsoft"**

**{}**
There are no documents that contain both the words Apple and Microsoft.

**q3 = "Apple OR HTC"**

q3DNF = (1, 1, 1) v (1, 1, 0) v (1, 0, 1) v (1, 0, 0) v (0, 1, 1) v (0, 0, 1)
sim(dj, q3) = (1, 1, 1, 1, 0, 1, 1, 0, 1, 1,)
**{doc1, doc2, doc3, doc4, doc6, doc7, doc9, doc10}**
These documents contain either the word Apple or the word HTC.

q4 = "Samsung NOT HTC"

q4DNF = (1, 1, 0) v (0, 1, 0)
sim(dj, q4) = (0, 1, 0, 1, 1, 0, 0, 1, 0, 0)
**{doc2, doc4, doc5, doc8}**
These are the only documents that contain the word Samsung, but NOT the word HTC.

q5 = "Apple"

q5DNF = (1, 0, 0) v (1, 0, 1) v (1, 1, 0) v (1, 1, 1)
sim(dj, q5) = (1, 1, 0, 1, 0, 1, 1, 0, 1, 1)
**{doc1, doc2, doc4, doc6, doc7, doc9, doc10}**
These are the documents that contain the word Apple.

2. What is the dimension of the vector space representing this document collection when you use the vector model and how is it obtained?
   There are 3 dimensions due to the existing of 3 expressions.

3. Calculate the weights for the documents and the terms using tf and idf weighting. Put these values into a document-term-matrix. (Tip: use the equations in the book and state which one you used.)

   TF:

| Doc | Apple | Samsung | HTC |
|---|---|---|---|
| 1 | 1 | 0 | 2 |
| 2 | 1 | 1 | 0 |
| 3 | 0 | 2 | 1 |
| 4 | 1 | 1 | 0 |
| 5 | 0 | 1 | 0 |
| 6 | 1 | 3 | 4 |
| 7 | 1 | 0 | 1 |
| 8 | 0 | 3 | 0 |
| 9 | 1 | 0 | 2 |
| 10 | 3 | 1 | 1 |

IDF:

| Begrep | ni | IDF=log(N/ni) N=10 |
|---|---|---|
| Apple | 7 | 0.51 |
| Samsung | 7 | 0.51 |
| HTC | 6 | 0.74 |

(1 + log(fi,j)) x log (N/ni)

| Wi,j | Apple | Samsung | HTC |
|---|---|---|---|
| doc1 | 0.51 | 0 | 1.47 |
| doc2 | 0.51 | 0.51 | 0 |
| doc3 | 0 | 1.03 | 0.74 |
| doc4 | 0.51 | 0.51 | 0 |
| doc5 | 0 | 0.51 | 0 |
| doc6 | 0.51 | 1.33 | 2.21 |
| doc7 | 0.51 | 0 | 0.74 |
| doc8 | 0 | 1.33 | 0 |
| doc9 | 0.51 | 0 | 1.47 |
| doc10 | 1.33 | 0.51 | 0.74 |

4. Study the documents 1, 2, 8 and 9 and compare them to document 4. Calculate the similarity between document 4 and these four documents according to Euclidean distance. (Use tf-idf weights for your computations).

*4 - 1:*
d = sqrt( (1-1)^2 + (1-0)^2 + (0-2)^2 ) = sqrt(5) = 2.24

*4 - 2;*
d = sqrt( (1-1)^2 + (1-1)^2 + (0-0)^2 ) = 0

*4 - 8;*
d = sqrt( (1-0)^2 + (1-3)^2 + (0-0)^2 ) = sqrt(5) = 2.24

*4 - 9:*
d = sqrt( (1-1)^2 + (1-0)^2 + (0-2)^2 ) = sqrt(5) = 2.24

5. **Rank the documents by their relevance to the query q5.**
   From 3.1.3 we have that the query q5 is represented by the vector (0.51, 0, 0) and for example d1 is represented as (0.51, 0, 1,47) in vector form. By calculating the cosine similarity using the following calculation:

$$\cos(q,d) = \frac{q \cdot d}{||q||||d||}$$

and ranking them by their relevance we get the table below:
Eks doc1:

$$\cos(q,d1) = \frac{0.51 * 0.51 + 0 * 0 + 0 * 1.47}{\sqrt{0.51^2 + 0^2 + 0^2} * \sqrt{0.51^2 + 0^2 + 1.47^2}}$$

| Doc: | Value: cos(q,dx) |
|---|---|
| doc2 | 0.707 |
| doc4 | 0.707 |
| doc7 | 0.567 |
| doc1 | 0.328 |
| doc9 | 0.328 |
| doc10 | 0.318 |
| doc6 | 0.194 |
| doc3 | 0 |
| doc5 | 0 |
| doc8 | 0 |

## *SubTask 3.2: Probabilistic Models*

*Given the following queries:*

*q1 = "HTC"*
*q2 = "Apple Microsoft"*

1. **Rank the documents using the BM25 model. Set the parameters of the equation as suggested in the literature. (Here we assume relevance information is not provided.)**

Setting K1 = 1.2 and b = 0.75, avg_doclen = 3.2.
BM25:

$$B_{i,j} = \frac{(K_1 + 1)f_{i,j}}{K_1\left[(1-b) + b\frac{len(d_j)}{avg\_doclen}\right] + f_{i,j}}$$

Okapi BM25 scoring:

$$sim_{BM25}(d_j,q) = B_{i,j} \times log_2\left(\frac{N}{df_t}\right)$$

Where $df_t$ is the number of documents the term exist in.

The results is shown in the table below:

| $sim_{BM25}(d_j, q)$ | q1 | q2 |
|---|---|---|
| doc1 | 1.03 | 0.53 + 0 |
| doc2 | 0 | 0.61 + 0 |
| doc3 | 0.76 | 0 |
| doc4 | 0 | 0.61 + 0 |
| doc5 | 0 | 0 |
| doc6 | 0.99 | 0.32 + 0 |
| doc7 | 0.87 | 0.61 + 0 |
| doc8 | 0 | 0 |
| doc9 | 1.03 | 0.53 + 0 |
| doc10 | 1.60 | 0.73 + 0 |

2. What are the main differences between BM25 model and the probabilistic model introduced by Robertson-Jones?
   The main purpose of the model introduced by Robertson and Jones is to serve as a framwork for future forms. The framework has multiple shortcomings which makes it unsuitable as an algorithm to weigh the documents. The algorithm isn't accurate enough to estimate the first round of probabilities and the terms are assumed mutually independent. BM25 is a collection of many scoring functions that weights documents with different formulas and parameters.