D5: Databases and Information Systems
Advanced Topics in IR, SS 2016
Dr. Vinay Setty and Dr. Jannik Strötgen
Assignment 1, due: **5 May 2016, 23:59:59**

max planck institut informatik

*Please submit your solution as a PDF and source code as a tar file to* `atir16@mpi-inf.mpg.de` *before the due date mentioned above!*

## Boolean Retrieval Model (20 points)

**Problem 1.** In the first lecture you got an introduction to boolean queries and how they can be processed using a binary matrix with terms as rows and documents as columns as an index. And the matrix contained 1s and 0s depending on if the term occurred in the document or not.

(a) What are the problems with using such a binary matrix as an index? What approaches can you think of to overcome these problems? Explain your solution with an example.

(b) One of the variants of boolean queries is to specify proximity of term occurrences. For example "Frodo AND Sam occurring within 10 words" from each other. Propose a solution to create posting lists for processing such positional queries. Explain your solution with an example.

## Language Models and Smoothing (20 points)

**Problem 2.** The first lecture included a quick recap of language models and smoothing methods. The likelihood of generating the query $q$ from the language model $\theta_d$ for document $d$ is

$$P[\,q \mid \theta_d\,] = \prod_{v \in q} P[\,v \mid \theta_d\,]^{\,tf(v,q)}$$

where the probability $P[\,v \mid \theta_d\,]$ can be estimated without smoothing as

$$P[\,v \mid \theta_d\,] = \frac{tf(v,d)}{\sum_w tf(w,d)}\ ,$$

using Jelinek-Mercer smoothing with the document collection $D$ as

$$P[\,v \mid \theta_d\,] = \lambda \cdot \frac{tf(v,d)}{\sum_w tf(w,d)} + (1-\lambda) \cdot \frac{tf(v,D)}{\sum_w tf(w,D)}\ ,$$

or using Dirichlet smoothing as

$$P[\,v \mid \theta_d\,] = \frac{tf(v,d) + \mu \cdot \frac{tf(v,D)}{\sum_w tf(w,D)}}{(\sum_w tf(w,d)) + \mu}\ .$$

Smoothing introduces a relative weighting of query terms similar to the inverse document frequency in tf.idf term weighting. Devise a **minimal example** (at most three documents) to demonstrate this effect.


## NLP-related questions (20 points)

**Problem 3.** In the second lecture, we discussed several NLP topics.

(a) As you already have some background in information retrieval, you might have been wondering that something important was missing: which (rather simple) NLP task(s) did we not cover although it is (they are) very crucial in the context of IR? Mention at least one NLP task and why it is so crucial for information retrieval. How could this task be addressed?

(b) Assume you want to build your own NER (Named Entity Recognizer) system that tags tokens as either *PERSON*, *LOCATION*, or *OTHER*. Unfortunately, you have no training data and you have to come up with some useful features. Then, you read the following article, which helps you a lot:

*In January, the man from Boston moved to the city of Paris. It is a beautiful city and compared to New York, London and Berlin, the whether is much better. At least, this is the impression of Klaus. When Hans called Paris, to tell her about his new job, she asked him if he forgot the words of George Washington: In Germany, Sarah is a common name but in the USA it is not.*

– Briefly discuss some general challenges of *named entity recognition*.
– List potential features in the form of simple rules that your system could use to determine whether a token should be tagged as *PERSON*, *LOCATION* or *OTHER*. A not very smart example is "if a token ends with 'l', it's tag is OTHER" (Why is it not very smart?).
– Provide brief explanations of your features using examples from the text.
– Is it possible to tag all tokens of the example text correctly using only your rule-based features? What's the main problem of using such a small piece of text to generate features?

(c) Independent of the Programming Assignment below, what are the advantages and disadvantages of using stemming versus lemmatization for IR? Given the example text above (everything lowercase). Stem the text using the Snowball stemmer (link to the demo is on the slides, page 21). How many "terms" do you have to put into an inverted index (i) if you use the text as it is (everything lowercase), (ii) if you apply stop word removal using the list shown in the lecture (slides, page 22)?

## Evaluation (10 points)

**Problem 4.** In the second lecture, we discussed several evaluation measures.

(a) Explain the problem of using *accuracy* to evaluate an IR system's retrieval quality? For which (NLP) tasks is accuracy a good measure and why? What's the main difference to precision?

(b) Assume you are not a student with a tiny laptop but an employee of a huge search company. Your boss tells you to evaluate the quality of your search engine. Convince her that all the evaluation measures you learned in the second lecture do not make much sense. Explain why you could do better if you are an employee of this huge search company and how you suggest to evaluate the company's IR system quality instead.


## Programming Assignment (30 points)

**Problem 5.** You are given an evaluation benchmark from the TREC (Text REtrieval Conference). It can be downloaded from `http://resources.mpi-inf.mpg.de/departments/d5/teaching/ss16/atir16/assignments/data/atir16-assignment1-data.tgz`. It is password protected by username "atir16" and password mentioned in the class (if you don't know ask your colleagues or send an email to atir16@mpi-inf.mpg.de).
The data contains three files described below:

(i) "documents.json" contains one json document per line, each with the following fields:

- id: document id
- title: title of the document
- pub-date: publication date of the document in UNIX epoch seconds format (optional, may be missing for some documents)
- content: body content of the document, this field still contains some original html-like markup such as  for paragraphs and  for headlines, ideally they should be removed.

(ii) "queries.tsv" contains a tab delimited file which has the following fields:

(a) query id/number
(b) query string/title
(c) query description (starts with a prefix "description:")
(d) query narrative (starts with a prefix "narrative:")

"description" and "narrative" fields can be ignored but they offer you the details on the information need, if you want to exploit this information you are free to do so.

(iii) "judgments.tsv" contains a tab delimited file which has the following fields:

(a) query id/number (same query ids as in queries.tsv)
(b) document id (same document ids as in documents.json)
(c) binary relevance (1 is relevant, 0 is not relevant)

In the lecture, you were given a brief introduction to elasticsearch.

(a) Using elasticsearch, index the documents in documents.json, process the queries in "queries.tsv", and using the relevance judgments given in the "judgments.tsv", measure the precision and recall for each query. In addition, calculate the mean average precision (MAP) using a cut-off of 10 (i.e., only ranks 1 to 10 are considered).
Try with different retrieval models such as boolean, TF-IDF and BM25 and discuss the changes you observe with respect to precision / recall or MAP.

(b) In the second lecture, you learned about NLP techniques such as stemming and stop word removal. Use a language analyzer similar to the one presented in the first lecture, consisting of stop word removal and stemming, and index the "documents.json". Then measure the changes in precision and recall. You may choose the best performing retrieval model from the previous task. Did you observe any improvement? Document your observations.