D5: Databases and Information Systems
Advanced Topics in IR, SS 2016
Dr. Vinay Setty and Dr. Jannik Strötgen
Assignment 2, due: **26 May 2016, 23:59:59**

max planck institut informatik

Please submit your solution as a PDF and source code together in a single tar/zip file, name your pdf file in the following format: "Lastname-Immatrikulationnumber-AssigmentX.pdf", also include Immatrikulationnumber in the title of your document and email it to `atir16@mpi-inf.mpg.de` *before the due date mentioned above!*

Logarithmic Merge for Search on Social Media (25 points)

Problem 1.
Read the paper by Wu et al. [9] in which they use logarithmic merge to deal with high arrival rates of posts (e.g., tweets) in social media.

(a) Explain their approach in your own words (a most one page $\approx$ 250 words).

(b) How could you adapt their approach so that only posts published within a specific recent period (e.g., the last month) are indexed and kept? Posts older than that should not be returned in query results and be pruned from the index.

(c) How could you adapt their approach so that it can efficiently retrieve all relevant posts published during a specific time interval $[t_b, t_e]$?

WAND-Style Query Processing with Static Scores (25 points)

Problem 2.
Assume that we want to rank documents according to a combination of (i) a static importance score $\text{imp}(d)$ (e.g., determined using PageRank) and (ii) a relevance score $\text{rel}(d)$ defined as

$$\text{rel}(q, d) = \sum_{v \in q} w_{tf.idf}(v, d)$$

with $w_{tf.idf}(v, d)$ as the *tf.idf* weight of term v in document d.

We now consider three ways how importance and relevance can be combined. For each of them, think about (i) how you can use WAND to efficiently determine top-k results, (ii) which posting lists you would keep in your inverted index, and (iii) which payloads postings in those posting lists would have.

(a) Linear combination of importance and relevance as

$$\text{score}(q, d) = \alpha \cdot \text{imp}(d) + (1 - \alpha) \cdot \text{rel}(q, d) \ .$$

Note that under this formulation a document could make it into the top-k only because of its high importance and without containing any of the query terms.

(b) Linear combination of importance and relevance as above. In addition, we only consider documents as potential results that contain *at least one* of the query terms.

(c) Combination of importance and relevance as product

$$\text{score}(q, d) = \text{imp}(d) \cdot \text{rel}(q, d) \ .$$

Jaccard Distance (10 points)

Problem 3.
In the lecture it was mentioned that a distance function d is a metric if it admits following properties:

- Non-negativity: $d(X, Y) \geq 0$
- Symmetric: $d(X, Y) = d(Y, X)$
- Identity: $d(X, Y) = 0$ iff $X = Y$
- Triangle inequality: $d(X, Y) + d(Y, Z) \geq d(X, Z)$

In the lecture we also defined the Jaccard distance function as:

$$Jdist(X, Y) = 1 - \frac{|X \cap Y|}{|X \cup Y|}$$

Prove that $Jdist$ is a metric.

Distributed Minhashing (10 points)

Problem 4.
In the lecture you learned how to compute minhash signatures for a given binary matrix representation of documents and shingles. Suppose we want to use a MapReduce framework to compute minhash signatures. If the matrix is stored in chunks that correspond to some columns, then it is quite easy to exploit parallelism. Each Map task gets some of the columns and all the hash functions, and computes the minhash signatures of its given columns. However, suppose the matrix were chunked by rows, so that a Map task is given the hash functions and a set of rows to work on. Design Map and Reduce functions to exploit MapReduce with data in this form. You can write the map and reduce functions as a pseudocode or you can also just describe what mappers and reducers do in your own words.

Programming Assignment (30 points)

Problem 5. In the last assignment you indexed a sample TREC corpus using elasticsearch. In this assignment your task is to implement a search result clustering. You can reuse the index from previous assignment to retrieve top-100 documents for the following queries.

1. Hubble Telescope Achievements
2. African Civilian Deaths
3. Implant Dentistry
4. Radio Waves and Brain Cancer

5. Alzheimer's Drug Treatment

- In the lecture you learned K-Means clustering technique. Implement the K-Means clustering to cluster the result documents for the above queries. You are free to choose any features for your documents (such as bag of words, sets, tf-idf vectors) and distance functions such as Jaccard distance or cosine similarity. Find the $K = 10$ clusters for the above queries. Compute the average point to centroid similarity for each cluster.
- Your next task is to select label for each cluster. A label can be a single word or a n-gram from the documents in that cluster that best describes the cluster. For example, for the query "Hubble Telescope Achievements", the labels "NASA programs" and "SETI" may represent two of the clusters. To avoid using the query terms as labels ignore the query terms as the candidate labels. You can use any features or techniques (such as tf-idf or Language Models) of your choice to select the label. List the labels you found for each of the queries. Additionally, you can also give higher weight to the potential named entities like the the words with their first letter capitalized. Verify manually if your label actually makes sense and give it a relevant judgment like relevant or not.

Tips:

- Elasticsearch also provides a way to get tf-idf vectors for each document, it can be directly used for computing distance metrics. You can do this by specifying "term_vector": "with_positions_offsets_payloads" in the mapping while creating the index, check the sample code for this.
- If you did not manage to successfully implement the first programming assignment, we provide simple program for indexing your documents and run queries in python. `http://resources.mpi-inf.mpg.de/departments/d5/teaching/ss16/atir16/assignments/data/assignment1-code.tgz`