D5: Databases and Information Systems
Advanced Topics in Information in IR, WS 2014/15
Dr. Klaus Berberich
Assignment 4, due: **8 Jan 2015**

Please submit your solution as a PDF to `atir2014@mpi-inf.mpg.de` *by the indicated due date!*

Effectiveness Measures for Novelty & Diversity

Problem 1.
Two retrieval systems return the results given in the table on the left for an ambiguous query.

	$\mathbf{R}_1$	$\mathbf{R}_2$
1.	d_1	d_4
2.	d_6	d_3
3.	d_2	d_7
4.	d_8	d_8
5.	d_5	d_1

	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9
a	0	2	0	1	0	2	1	0	2
b	1	0	1	0	1	0	2	1	2

We further know that there are two query aspects **a** and **b** (each equally popular among users) and have collected the graded relevance assessments given in the table on the right.

(a) Compute standard nDCG for the two retrieval results. Use the maximum of the two graded labels assigned to a document for the two query aspects as its unified graded label.

(b) Compute intent-aware nDCG (nDCG-IA) for the two query results.

(c) Compute α-nDCG ($\alpha = 0.5$) for the two query results. Use query aspects as information nuggets and treat documents with a graded label in $\{1, 2\}$ as relevant.

Submodularity

Problem 2.
Carbonell and Goldstein [3] describe Maximum Marginal Relevance (MMR) as a greedy selection rule. Analogous to IA-Select by Agrawal et al. [1], we can alternatively cast MMR into the following optimization problem

$$\arg\max_{S \subseteq R} \sum_{d \in S} \left(\lambda \cdot \text{sim}(q, d) - (1 - \lambda) \cdot \max_{d' \in S} \text{sim}(d, d') \right) \quad \text{s.t.} \quad |S| = k$$

where R is the set of all documents, q is the query, and S denotes the selected set of k documents.

(a) Is the objective function of the above optimization problem *submodular*? Prove your answer.

(b) Does the greedy selection rule given in [1] thus provide an approximation guarantee?

Maximum Marginal Relevance (Programming Assignment)

Problem 3.
On the course website you can download all articles published by The New York Times in June 2002 (`200206.tar.gz`). There is also a document (`nytimes-corpus-overview.pdf`) describing the data format and, if you want to use Java, a library (`nyt-tools.zip`) providing a parser for the documents. We now want to implement a small-scale in-memory search engine over this data and compare the results obtained by MMR for different choices of λ.

(i) Parse the documents, extract the text from the `body` field, convert it to lower case, and tokenize it by splitting at all non-alphanumeric characters (i.e., `[^a-z0-9]`), use the `guid` field as a document identifier, and also keep track of the URL from the `url` field.

(ii) Compute *tf.idf* vectors for the documents using the following *tf.idf* variant

$$w_{tf.idf}(v, d) = tf(v, d) \cdot \log \frac{|D|}{df(v)} ,$$

normalize the vectors (so that $\|\boldsymbol{d}\| = 1$). Build an *inverted index* (e.g., using a hashmap) that allows you to retrieve all vector components for a specific term. Build a *direct index* that allows you to retrieve all vector components for a specific document.

(iii) Implement Maximum Marginal Relevance (MMR). As a first step, determine the similarities $\text{sim}(q, d)$ for the given query q (using binary component weights for the query vector). These documents constitute the set R from which you now select the subset S. The first document to be included in S is the one having highest $\text{sim}(q, d)$. Now, include more documents in S using the greedy selection rule and computing $\text{sim}(d, d')$ using the precomputed normalized vectors.

(iv) Determine the top-5 results for the queries world cup, brazil, grammy award, and kashmir using $\lambda \in \{0.1, 0.5, 0.9, 1.0\}$. Please include the rank and the URL for each result document in your submission.

Logarithmic Merge for Search on Social Media

Problem 4.
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 ≈ 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 (Optional)

Problem 5.
Assume that we want to rank documents according to a combination of (i) a static importance score imp(d) (e.g., determined using PageRank) and (ii) a relevance score 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

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

Note: Given that we did not manage to cover WAND and Block-Max WAND in the lecture, this problem is optional. Feel free to attempt it, but it does not count toward the 50%.