Lab Project

Coordination

- Check your mails regularly
- Join our Slack group: irlecture.slack.com
  (invitation link sent in an earlier mail)
- Ask questions via Slack, not via mail, so that others can follow up, too
- Coordinate with others on shared problems
- Organize your material on our GitLab
  Groups corresponding to the aforementioned ones have been set up, but not all of you have an account, yet
- Take a look a previous projects collected there

- Need (more) data?
- Need access to lab?
- Need access to cluster?

Lab Project

First steps and Priorities

1. Search related work and existing search engines
   Google Scholar, Semantic Scholar, Microsoft Academic

2. Get the data and analyze it
   (e.g., What are important characteristics? Plot variables of interest.)

3. Build a vertical prototype with defaults
   (e.g., Apache Lucene, Apache Solr, Terrier, Vespa)

4. Build a web interface to access the search engine

5. Make sure the web server logs all interactions
   (at least a user's IP, queries, result clicks, and dwell times)

6. Build an evaluation environment

Lab Project

Milestones

- **Data Acquisition** Download and understand the data format; preprocess the data.
- **Data Analysis** Descriptive statistics, highlighting variables of interest.

- **Technology Stack** Decide upon the software libraries you are going to use.
- **Vertical Prototype** Working prototype with a basic retrieval model.
- **Refined Prototype** Prototype that uses and advanced/refined retrieval model.

- **Evaluation Topics** Collect topics relevant to your search domain: target 50.
- **Relevance Feedback** Implement logging and relevance feedback facilities.
- **Relevance Judgments** Carry out relevance judgments for the topics.

- **Deployment** Ensure that your software can be started easily, e.g., Ubuntu 18 + Docker.
- **Documentation** Write a README; incl. all commands for deployment.
- **Report** Write the final report of up to max. 10 pages.

Lab Project

Evaluation: JSON Schema

Topic:

```
1. {
2.   "topic_id": string (Some unique ID; probably just a number.),
3.   "query": string (The query submitted to the search engine.),
4.   "description": string (Description of the user's intent.),
5.   "narrative": string (Specification of what relevant documents are.)

6. }
```

Relevance judgment:

```
1. {
2.   "topic_id": string (Reference to the topic.),
3.   "document_id": string (Reference to the document judged.),
4.   "relevance": number Graded relevance judgment, e.g., {0,1,2}.)
5. }
```

Make sure that document IDs are permanent.

When reindexing from scratch, they must not change.

Lab Project

Evaluation: JSON Example

Example topic:

```
1. [
2.   {
3.     "topic_id ": "t42",
4.     "query": "black bear attacks",
5.     "description":
       "To find the frequency of black bear attacks worldwide and
       possible causes for this savage behavior.",
6.     "narrative":
       "It has been reported that food or cosmetics sometimes attract
       hungry black bears, causing them to viciously attack humans.
       Relevant documents would include the aforementioned causes as
       well as speculation preferably from the scientific community
       as to other possible causes of vicious attacks by black bears.
       A relevant document would also detail steps taken or new
       methods devised by wildlife officials to control and/or modify
       the savageness of the black bear."
7.   },
8.   ...
9. ]
```

Lab Project

Evaluation: JSON Example

Example relevance judgments:

```
[
  {
    "topic_id": "t42",
    "document_id": "d051",
    "relevance": "0"
  },
  {
    "topic_id": "t42",
    "document_id": "d053",
    "relevance": "2"
  },
  ...
]
```