Resources for the practical projects

Here, we will provide resources that might be helpful for the practicals.

Dataset Location

Feel free to use (or create) your own dataset.

Below, you find a list of links to get you started.

Specific datasets:
- https://www.kaggle.com/datasets/stackoverflow/statsquestions (Stack Exchange, question/answer pairs)

Benchmark collections:
- https://huggingface.co/BeIR (IR benchmark, often large)
- https://huggingface.co/mteb (Embedding benchmark, not only IR)

Websites with easy to use APIs:
- https://pypi.org/project/wikipedia/ (Wikipedia)
- https://github.com/pushshift/api (Reddit)

Popular dataset repositories/catalogs/search:
- https://huggingface.co/datasets
- https://www.kaggle.com/datasets
- https://paperswithcode.com/datasets
- https://ir-datasets.com/index.html
- https://datasetsearch.research.google.com/

Standard IR test collections (as discussed in the lecture, but these are often rather large):
- https://trec.nist.gov/data.html
- https://www.clef-initiative.eu/

Example Project Ideas

Here, is a non-exhaustive and non-systematic list of ideas for the project.

1. Retrieval performance on summarized documents.
   > Automatically summarize documents. Compare retrieval performance with non-summarized documents.
2. Retrieval performance on translated documents.
   > Automatically translate documents. Compare retrieval performance with non-summarized documents.

-> 1. and 2. could even be combined.

3. Debiased retrieval performance.
   > Measure a bias (e.g., toxicity). Create a model that factors in the bias (e.g., lower weight). Compare performance to non-debiased baseline.

-> Good domain for 1.,2.,3. would be news for instance.

4. Product ranking based on the descriptions in a few-shot setting.
   > Define a metric for ranking (e.g., product quality). Train on pairwise rankings with the description as input. Test on previously unseen items. Should leverage knowledge from language models (e.g., transformers) for exploitation on novel (unranked) items.
5. Similarity of words (e.g., programming languages).
   > Train a Word2Vec model on programming languages. Find similarities and analogies.
6. Evaluate a neural IR system.
   > Use pretrained models. Evaluate time to index documents and retrieval, as well as performance. Comparison to non-neural models.
7. Content analysis (e.g., on misinformation/conspiracy theories).
   > For instance with topic modeling and clustering of words/documents to find over-represented words.

8. Reproduce IR algorithms from a paper.
9. Look at challenge websites. For instance: TREC, Kaggle (but check that its focus lies on IR).

Other valid tasks, but not explicitly covered in class:
- recommendations
- question answering
- conversational systems
- query expansion
- categorization/classification (including sentiment analysis/opinion mining)
- domain-specific IR
- multi-modal IR (e.g., retrieving music or images by, e.g., wave or timm models)
- pure NLP tasks (e.g., translation model, information/entity extraction)
- explicit behavioral modeling (e.g., repeat consumption patterns, predict conversion rate/click-through rate)

Other resources

Free GPU platforms:
- Colab
- Kaggle
- Paperspace
- DeepNote but only has CPU, no GPU in free tier

Literature:
- Transformer for IR Paper

Online materials:
- Python4DS Handbook
- Pytorch Tutorial
- HuggingFace Course
- Transformer Illustrated
- sBERT Libary Documentation