Final Project: Text Retrieval by Content, Image Retrieval by Content or Article Recommendation

Instructor: Xiaoming Jin
xmjin@tsinghua.edu.cn
Teaching Assistant:
Zihan Zhang: zhangzihan.zac@gmail.com
Changsheng Xiang: xcszbdnl@gmail.com
Due date: May 20 (proposal), June 7 (final report)

Three most important things:

- Before you start this project, please read this document carefully, and make sure your submission meets all requirements.
- Intra-group collaboration is encouraged, but inter-group "collaboration" in this project is not permitted.
- All the materials provided by this project are not permitted to be redistributed for any purpose.

1 Introduction

For the final project, we provide three tasks, namely Text Retrieval by Content, Image Retrieval by Content and Article Recommendation by Content. You can choose any **ONE** according to your own interest. However, here are some aspects you should take into account before you make your decision.

- Scoring will be based on the difficulty of tasks: we will be more strict and cautious for the easier task.

- Here, the order of the tasks (from easiest to hardest) is: Text Retrieval, Article Recommendation, Image Retrieval.

All tasks are open projects in which you could try any effective techniques in the context of this course; moreover, you can refer to the Internet for more related materials. Creativity is encouraged in your work!

The rest of this document is organized as follows: Section 2 presents specification for Text Retrieval by Content in detail. Section 3 specifies details for Article Recommendation by Content task. Section 4 presents the task of Image Retrieval by Content. Requirements for a paper reading and discussion session are described in Section 5. Section 6 lists project submission materials and related policy. Some useful hints and resources are provided in section 7, which you can can skip if you have already had sufficient prior knowledge about the task.

2 Text Retrieval by content

This section presents the task of Text Retrieval by Content, if you have decided to do the other task, just skip or read this section for fun.

2.1 Overview

Text retrieval is defined as the matching of some stated user queries against useful parts of free-text records [1]. It has attracted increasing interests from both research and industry fields due to its importance in many real world applications, e.g., Web search and mining, digital library, etc.

Traditional search engines or text retrieval systems are mainly based on keyword queries. In this setting, users provide one or more keywords as “query”, and documents containing the given keywords are retrieved from database as search results. In the past decades, there have been increasing interests in text retrieval by content, with many new techniques developed in this field [2]. The query-by-example problem is an instance of text retrieval by content: a document is provided as query and the documents whose contents are most similar to the query should be returned as results.

This project gives you an opportunity to build a content-based text retrieval system. The goal is to improve user experience in finding documents of his/her interests, or more specifically, documents that are similar to given queries.

2.2 Task Specification

This project includes two required and one optional parts, as well as some paper readings, a discussion session and a presentation session(Refer to Section 5 for details.).

Part I: Build a system for the fundamental retrieval task. (Required) You are given a collection of 14,035 documents, each of which is a plain text file. You are asked to build your own text retrieval system for the query-by-example task. In this task, a plain text will be the input as user query, and a list of k ($k = 10$ for this project) relevant documents, whose contents are most similar to the query, should be returned as the result.

Part II: Improve the performance of your system. (Required) In addition to the standard retrieval system mentioned above, you should try some more sophisticated approaches which may further enhance user experience. You are required to choose **ONE** option in the following list to improve your system. Note that the difficulty varies among the listed options. If you choose relatively easier ones, you will be expected to finish them with a better performance.

- Relevance feedback [3]. You can try to collect feedback from users to further capture the intent of the user and enhance the system performance.

- Search results clustering [4]. This may be useful as a navigation tool for users to browse the search result. See [5] for an example.

- Dimension reduction. Since feature vectors for text collection normally have very high dimensionality, you may consider some dimensionality reduction techniques, such as SVD/LSI [6].

- Document summarization. You can try to generate a piece of summarization for each returned document. This summarization can be used to speed up search experience by enabling users to judge whether the document is relevant or not before they click on it, which is now widely used by almost all web search engines.

- Implement other query mechanisms. Except for the vector space model, you can also implement the boolean model (mainly for keyword search) or probabilistic model or any other models that you believe it would work well for the retrieval task. See relevant chapters from [7] for reference.
- Support "advanced" keyword search in addition to query-by-example scheme. Keyword search might be another good starting point when searching a large text database. Note that, however, if you simply transform the keywords into a vector and use the original query-by-example scheme, you may find that the performance of search system will not be very satisfactory.
- Personalized search service. To do this, you may first collect some historical data by a certain user and then analyze how these data can be used to adapt search results to a specific user.
- Query both text data and image data. Most commercial search engines will also list some relevant images as the search result. You can try to crawl some data and support this in your system.
- Any other techniques that may help improve the performance and/or efficiency of your system. There is a rich literature in this context, which might be helpful for you.

Optional Part: Choose another option from the above list, or try any other ideas you deem useful to further improve your system. We strongly encourage you to try your own ideas. This part of work could help you obtain a higher score (up to 10% in grading with a 110% ceiling in total) for this project. But note that, 1) the optional part cannot be used as a replacement for the required part; 2) any improvement that is unrelated to this course will not be considered in grading.

Result File Format: You should save the search results into a single plain text file with the following format:

- Search results for all queries should be stored in this file.
- The result file should contain 5000 lines, corresponding to the 5000 queries, where the n-th line corresponds to the search result for the n-th query.

```
100:10015,2014,4139,4116,1383,11511,11504,20889,209,20863
200:20882,14064,11108,11127,10001,21988,2197,18340,18330,18346
......
```

Table 1: An example for the result file.

- In each line, the first term is the query, the following terms are the results of this query. The query and the results are separated with a colon. The results consist of 10 comma-separated (just one single comma and no spaces) numbers, which are the IDs (filenames) of the returned texts, sorted in descending order of their similarity values to the query. The first number corresponds to the ID (filename) of the most similar document to the query; the second number corresponds to the ID of the second most similar document; and so forth.
- Note that any other formats will **NOT** be accepted.

Table 1 is an example of the result file.

Evaluation: We will provide 5,000 test queries right before the presentation session (see Section 5 for detail). When your group make presentation, your group will be asked to run these test queries and produce a result file. You should submit your search results for all these queries to the course website before deadline. Then these results will be evaluated by teaching assistants manually and subjectively. Besides, as no evaluation query is provided until presentation session, you have to use parts of the provided documents as queries to evaluate your system performance by yourself.

3 Article Recommendation by Content

This section presents the task of Article Recommendation by Content, if you have decided to do the former task, just skip or read this section for fun.

3.1 Overview

Recommender system is a specific type of information filtering (IF) technique that attempts to present information items (movies, music, books, news, images, web pages, etc.) that are likely of interest to the users [8]. With the rapid growth of information on the Internet, it has been increasingly important and challenging for users to get the information conforming to their needs. Generally Speaking, there are 4 main approaches for recommendations [9] :

- Personalized recommendation: recommend things based on the individual's past behavior
- Social recommendation: recommend things based on the past behavior of similar users
- Item recommendation: recommend things based on the item itself
- A combination of the three approaches above

This project provides you an opportunity to build a recommendation system based on the textual articles. The goal is to recommend articles that match the users' historical interests.

3.2 Task Specification

This project has only one required task, as well as some paper reading, a discussion session and a presentation session (Refer to Section 5 for details.).

Training Data The training data consists of 2 files which contains 5551 users and 16980 articles in all. The information for each file is:

- `user-info-train.txt`: This file provides the collection history of users. Each line with a <user_id, article_id> pair format corresponds to a record of a specific user. Take the first line (1,496) for instance, it indicates that user with id number 1 has collected article with id number 496. For each user, at least 5 articles and at most 202 articles are collected.
- `raw-data.txt`: This file contains raw content for each article including article title and abstract. Each line is in the form of <article_id, "article_title", "rticle_abstract">.

```
7,3293;3918;6724;7496;8645
```

Table 2: An example of result file.

Test Data The test data consists of the same 5551 users as in the training data. For each user, 251 candidate articles are provides for prediction. Among these 251 articles, at least 5 articles have been collected by the user, others are sampled randomly. The test file is:

- `user-info-test.txt`: The form of this file is the same with `user-info-train.txt`.

Task Description The required task of this project is to build a recommendation system for fundamental article recommendation based on the collection history information provided in file `user-info-train.txt` and content information of each article provided in file `raw-data.txt`. You need to select 5 articles from the 251 candidates in `user-info-test.txt`, which mostly meet the user's interests.

You should write the recommendation results into a single plain text file with the following format:

- Recommendation results for all users should be stored in this file.
- The result file should contain 5551 lines, corresponding to the 5551 users, where the n-th line corresponds to the recommendation result for the n-th user.
- Each line consists of 5 article IDs separated by ';'. The IDs are sorted in descending order of their probability to be collected by a user. The first number corresponds to the user ID and we separate it with article IDs by comma. The form for one line in the result file is shown in Table 2.
- Note that any other formats will **NOT** be accepted.

Evaluation: During the presentation session (See section 5 for details), when your group make presentation, your recommendation results will be evaluated by our `EvalPred()` function at the same time. By this function, the Average Precision(AP) [10] measurement will be calculated. And scores are given based on the rank of AP@5, which means the group with the highest AP@5 precision will get the highest score. Please note that it is TA's job and there is no need for you to evaluate the results in the presentation session.

Because the result file only contains the top 5 recommended articles, so the AP@5 is calculated as below:

$$AP@5 = \frac{\sum_{k=1}^{5} P(k) \times rel(k)}{\texttt{number of all collected articles}} \tag{1}$$

where rel(k) is an indicator function equaling 1 if the article at rank k is a relevant article, zero otherwise. And P(k) is the precision at cut-off k in the list. For more information about AP@5 measurement, refer to [10] and [11].

4 Image Retrieval by content

This project gives you an opportunity to build a content-based image retrieval system. The goal is to improve user experience in finding images of his/her interests, or more specifically, images that are similar to given query.

- This task and the image dataset are the same as Project 1.
- However, you should utilize any other techniques instead of R-tree.
- The result file should adopt the same format as the Text Retrieval task.

We will provide 2,000 test queries before the presentation session. Your group will be asked to run these test queries and produce a result file at the presentation session. You should submit your search results for all these queries to the course website before deadline. Then these results will be evaluated by teaching assistants manually and subjectively. Besides, as no evaluation query is provided until presentation session, you can use parts of the provided images as queries to evaluate your system performance by yourself.

5 Discussion and Presentation Session

Paper reading and discussion session: A paper reading and discussion session are required for all these three tasks. The discussion session will be arranged at the end of May, 2016. Before this session, each group is required to submit a proposal in `pdf` format before May 20, 2016. This proposal should be limited to 4 pages and include:

- Related work: You should read at least **5** published papers related to the task you have chosen and line out the main ideas in these papers. Some analysis about the weak points and strong points about these proposed methods are also encouraged.
- General design of your system and your key ideas to implement this project.

We will select some interesting, not necessarily the best, proposals to make presentation in the discussion session. Each selected group needs to prepare a `ppt` (or `pdf`) to present your proposal in the discussion session. Moreover, you may also offer comments to other groups and help them improve the system during the discussion session. This part will also be considered in the grading.

Presentation session: The presentation session will be arranged in the last class of this course (June 7, 2016). In this session, each group should recommend one member to give a short presentation for your system (within 6 minutes), including its implementation, characteristics and performance.

6 Submission

6.1 Submission materials

Each group should submit the following materials before May 20, 2016:

- A `proposal` with pdf format. The proposal should include the summarization of at least 5 related papers to the chosen task as well as your own ideas. All the contents should be limited to 4 pages.

Right after or at the presentation session (June 7, 2016), each group should submit the following materials:

Table 3: Summarization of All Tasks

	Text Retrieval	Article Recommendation	Image Retreival
Grading	110% ceiling	110% ceiling	110% ceiling
Has optional or not	Yes	No	No
Evaluation	Manually evaluation	By AP@5	Manually evaluation
Discussion	Required	Required	Required
Presentation	Required	Required	Required

- A `technical report` that describes your system as well as the experiments. The report should be limited to 3 pages using **the given template** [12], which can be found in the `template` folder. **If not using template**, 10% ceiling for overall score will be penalized. Both LaTeX and Microsoft Word templates are provided. Please convert your report to `pdf` format for submission. The content of the report should include but is not limited to
  - your key ideas and main considerations
  - the design of your system and the key techniques you use
  - main results and conclusions you get.
  - In your report, techniques or packages borrowed from public resource should be explicitly indicated with necessary references.
- The `result file` produced by your system: 1) For the text retrieval task and the image retrieval task, it is the retrieval results for the query examples provided before the presentation session. 2) For the article recommendation task, it is the recommendation results file.
- The `source code and executable files`. You can use any coding language to finish the tasks. A `README` file containing detailed configuration of your system such as compiling environment, the operating system version, etc. should also be submitted with the source code.

6.2 Grading Policy

The final score of the three projects will be mainly based on the performance or accuracy of your system, improvements you have achieved, your technical report, and your oral report at the presentation session. The technical evaluation will include the user experience, technique strength, and novelty. For the Text Retrieval task and Image Retrieval task, we will mainly focus on the meaningfulness of your results, e.g. how relevant the search results are to the respective query. The efficiency of your system will also be considered. For the Article Recommendation task, your rank of AP@5 is the most important.

6.3 Important Dates

- Submission due of discussion proposal: 23:59:59, May 20, 2016.
- Submission due of technical report, source code and experimental results: 23:59:59, June 7, 2016.
- Presentation and system demo: June 7, 2016, in class.

All the materials should be submitted to the course website. Please note that late submissions will **NOT** be accepted.

6.4 Key Points

- You are allowed to have at most **THREE** members in each group. As announced before, intra-group collaboration is encouraged, but inter-group "collaboration" will lead to immediate failure.
- For the presentation session, you should prepare your programs in advance, and make sure it can work and output the results. You can use either your own computer or the computer in classroom to present your work.
  - If you choose to use the computer in classroom. For Text Retrieval task, we will copy the text data to C:\text\base and test query to C:\text\query in advance. For Image Retrieval task, we will copy the image data to C:\image\base and test query to C:\image\query in advance. For Article Recommendation task, we will copy the training data to C:\recommendation\train and testing data C:\recommendation\test. TA will have JRE and .Net framework installed on that computer. If particular system setting or

environment is needed for running your program, please contact TA before June 7, 2016.

  - If you choose to use your own computer. Please copy all the files mentioned above to your computer before class and make sure your program works well because you are not allowed to debug your program during the presentation.

7 Hints and Resources

You may find the following open source packages for text pre-processing helpful:

- The bow package [13] is a library of C code useful for writing statistical text analysis, language modeling and information retrieval programs. You can compile and use it under Linux environment. To start with, you can first try rainbow [13], which is a popular and easy-to-use text mining user interface based on the *bow* library.
- The Word & Web Vector Tool package [14] is a flexible Java library for statistical language modeling. It supports the creation of word vector representations of text documents in the vector space model.
- The Text to Matrix Generator (TMG) [15] is a MATLAB toolbox that can be used for various tasks in text mining. It can convert texts into a term-document matrix and supports techniques like stemming, normalization, stop word removal and different weighting schemes. Note that you need to fill in a request form online in order to get the package.

For the AR task, other resources are listed as follows:

- The **Apache Mahout** [16] machine learning library is to provide scalable implementation for several classical machine learning algorithms, e.g., clustering, classification and collaborative filtering.
- The **GraphLab** [17] collaborative filtering library is an efficient probabilistic matrix factorization library. In this library, multiple matrix decomposition algorithms are implemented.

- The **Caffe** [18] is is a machine learning framework that supports efficient learning using deep structure.

Please read the documentations of these packages for detailed instructions. Note that you are free to choose any tools, as long as you use the packages under their licenses.

References

[1] http://en.wikipedia.org/wiki/Document_retrieval/.

[2] C. Faloutsos and D.W. Oard. A Survey of Information Retrieval and Filtering Methods. 1998.

[3] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *JASIS*, 41(4):288–297, 1990.

[4] Hua-Jun Zeng, Qi-Cai He, Zheng Chen, Wei-Ying Ma, and Jinwen Ma. Learning to cluster web search results. In *SIGIR*, pages 210–217, 2004.

[5] http://search.yippy.com/.

[6] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[7] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[8] http://en.wikipedia.org/wiki/Recommender_system/.

[9] http://www.readwriteweb.com/archives/recommender_systems.php.

[10] http://en.wikipedia.org/wiki/Information_retrieval.

[11] M. Zhu. Recall, precision and average precision. *Department of Statistics and Actuarial Science, University of Waterloo, Waterloo*, 2004.

[12] http://www.acm.org/chapters/policy/toolkit/template.html.

[13] Andrew Kachites McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/∼mccallum/bow, 1996.

[14] http://wvtool.sourceforge.net/.

[15] http://scgroup20.ceid.upatras.gr:8000/tmg/.

[16] http://mahout.apache.org/.

[17] http://graphlab.org/.

[18] http://caffe.berkeleyvision.org/.