600.466 – Information Retrieval and Web Agent Project Report

Jinyi Guo (jguo32) and Qing Cai (qcai3)

May 13, 2016

1 Overview

In this project, We builded a resume finder robot (Option 3) which can implement three functions: (1) find resumes on the web (restricted to the JHU domain), (2) extract information into structure data format, and (3) find similar resumes if given a query resume. To analyze these resumes, We applied text rule-based classification techniques to segment resume sections (i.e. contact inforamtion, education, research, employment, teaching, publication, student, skill). Here we let all academic achievements such as papers, books, presentations, and patents into the segment of publication. We then extracted the key information (i.e. name, email, phone, address, C++, Java, degree) into a structured database format. Finally we wrote a cosine similarity function to calculate similar measuremnt between the query resume with each found resume and then found out the top similar resumes. In summary, our robot can search for resumes in JHU domain, extract key information and find the resumes we want.

2 Procedure

The more detailed procedure are shown as follows.

1. Begin with the existing web robot we used in HW 4.
2. Add a function to the robot's code to visit students' and faculties's homepages where we suppose the resumes exist.
3. Add a function to the robot's code to find resumes on the currently-visited homepage.
4. Extract the texts from the resumes which are mostly in .pdf format.
5. Add a function to segment resume sections (i.e. contact inforamtion, education, research, employment, teaching, publication, student, skill) by the rule-based method as in HW 1.
6. Extract the key information (i.e. name, email, phone, address, C++, Java, degree) into a structured database format.
7. Add a similarity function which can give a similarity measurement for two resumes as in HW 2 and 3.
8. Output the top similar resumes to the input resume or key words based on the similarity measurements as in HW 2 and 3.

3 Project Structure

Our project contains two major parts: web robot and text analyzer, which are included in file `pdf_robot.pl` and `segment_classify_similarity.prl` respectively.

To maximize the performance of both sides, we separated the parts instead of plugging the text analysis into the web robot. For the robot/crawler, we set the delay time to zero to maximize speed, though at a

risk of adding unexpected traffic to the server. We are sure to get exactly the same result if we set the delay time to up to 1 minute, but with considerably more running time.

Since the environment of folder `project` is exactly our development environment, one can test the program by the following commands:

```
perl pdf_robot.pl mylogfile.log content.txt http://www.cs.jhu.edu
```

and

```
perl segment_classify_similarity.prl.
```

4 Evaluation

Due to time limitation we only labeled one resume to test the performance of our classifier and information extractor. There are about 330 lines in the resume we labeled, for each line we judge which category (e.g. Education, Research and Publication etc.) it belongs to and label it manually. Then we test our classifier by comparing the classification result with the pre-labeled file. Here is the evaluation result:

```
OVERALL CORRECT: 283 = 85.2409638554217% INCORRECT: 49 = 14.7590361445783%.
```

For the information extraction part, since it is difficult to evaluate it performance by accuracy, we just went over some of the results to check if they are reasonable. Here are some example of key information we extracted from two CS professors:

```
Yanif Ahmad
Address:  Baltimore, MD 21218
EMAIL: yanif@jhu.edu
PHONE: (410)-516-6781
Degree:  Ph.D.
Randal Burns
Address:  Baltimore, MD 21218
EMAIL: randal@cs.jhu.edu
Degree:  Ph.D.
```

Note that we also implemented the function that compute the similarity between the query resume and all resumes in our resume pool. We did not have enough time to test its performance using labeled file, but we can get a rough feeling of the performance by using the resume of some professors we know. Here is the result of setting the resume of Prof. Randal Burns as query resume to find the similar resume in our pool (with about 150 resumes from all across JHU):

```
./resume/cv-yna.pdf
Yanif Ahmad
Address:  Baltimore, MD 21218
EMAIL: yanif@jhu.edu
PHONE: (410)-516-6781
Degree:  Ph.D.
Similarity:  0.740348219739564
Rank:  1
./resume/http:..www.cs.jhu.edu. mrg.cv.gormley-cv.pdf
Matthew R. Gormley
Degree; Ph.D.
Similarity:  0.621221528539565
Rank:  2
./resume/Yair_resume.pdf
Yair Amir
```

```
EMAIL: yairamir@cs.jhu.edu
Degree; Ph.D.
Similarity:  0.579043747929995
Rank:  3
```

Notice that all these three people are CS professors as well, and two of them share a lot of common research focus with Prof. Randal Burns.