CS 410 Progress Report – Automatic Crawler of Faculty Pages

Net IDs: ap41, gangyao2, sg54

Group Name: Team Texas

Scope in the proposal document.

1. Develop an automatic crawling tool that grabs URLs from a given university web page.
2. Train a classification model with a training instance for determining if a URL corresponds to a faculty directory page.
3. Identify the URLs corresponding to faculty member biographies.
4. Validate the classification models.

We have been working on points number 1 and 2. Below we include a detailed description of what has been accomplished regarding those 2 tasks.

Which Tasks Have Been Completed?

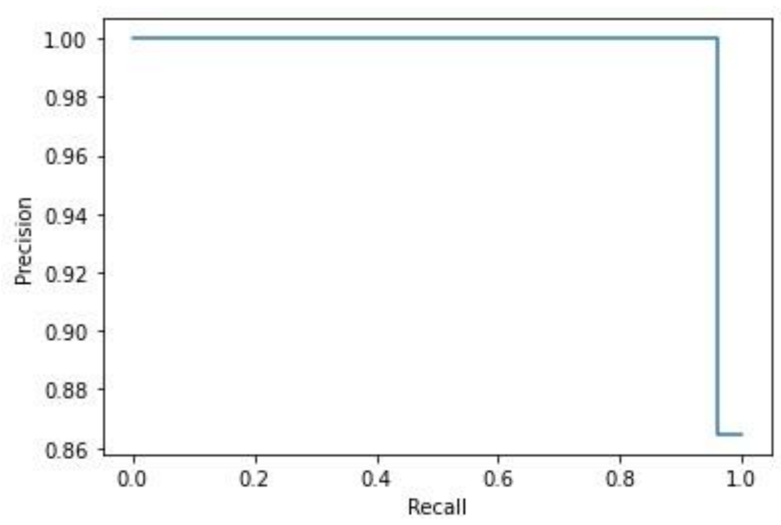
For the training of our model and model validation:

We implemented a URL based topic binary and URL content-based classification. We denoted 1 for faculty directory and 0 for non-faculty directory.

We focused content-based classification as using just a URL provides very little information for classification. The basic idea is to scrape the website using the Beautiful Soup and then extract all the text information from the site. Once the text data is extracted, then use *Apriori Algorithm* and set the Minimum Support to 50 to reduce the vocabulary size. We built 340 training datasets with the correct labels for each URL and use these training datasets to train the *SVM classification model*. This model is pretty accurate as the input has enough information for classification. However, the problem of content-based classification is scalability and efficiency. First, the training time is pretty long because we need to scrape every URL for the training dataset and extract its text information. Second, the validation time is long as well because we need to scrape the testing dataset URL.

Due to the performance problem of content-based classification, we switched to implementing URL topic classification. For this methodology, it doesn’t need to scrape every URL and extract the web text information, so the runtime is very fast. The drawback of this method is that there is very little information in the URL itself. In order to resolve this issue, we can use supervised tokens and *n-grams* derived from the URL to generate the features for classification.

We built a *SVM classification model* based on this concept and validated the model with another 229 testing datasets. The average precision of the model is 0.96 and the precision/recall curve is shown as the following.



For feeding the model from an automatic crawler:

We implemented a crawler using the ***scrapy framework***. So far, this spider has the following characteristics that allow us to experiment with how the URLs are produced:

- A *LinkExtractor* that ignores most file extensions like pdf’s, images, etc.
- A filter in the *LinkExtractor* is configured to only consider a domain and its subdomains. The crawler will not drift to other places on the internet.
- The Crawler can be switched from *BFO* to *DFO* exploration so we can measure the most effective way to get potential *URL* candidates.
- A configurable *depth-limit* for the crawling process. We estimate that given the main page of a university, the candidate *URLs* should be available at a relatively shallow level.
- Established a pipeline to process the URLs yielded by the Spider, concretely, ask a classification model to decide on the URLs.

Which Tasks are Pending?

For the training of our model:

Train additional classification models for the remaining tasks:

- Given a set of candidate URLs, refine such set considering the content of the page. This refinement of the set might not be required depending on the evaluation of the results.
- Apply similar models used for finding the directory pages to also find biography pages.
- Further testing and validation of our model.

Feeding the model from an automatic crawler:

- Refine the crawler settings as we start measuring the best way to yield URL's to the classification models.
- Enrich the pipeline process to include classification models or any other processing step.
- Further testing and validation of the integrated crawler + model.

Provide a user interface:

- In order to have a cohesive experience with our project, we want to provide an easy-to-use user interface which allows users to submit custom university URLs and get the results from the crawler and classifier.
- As of now this has not been worked on much as it relies on the full integration of the crawler and the classifier but will be started soon.

Are We Facing Any Challenges?

1. By performing classification using only URLs we believe that we will have a considerable number of false positives. One idea we have is to refine that list of URLs by using a second classification model based on the actual content of pages. Note that such a model would not run in a realistic time if applied to all the URLs yielded by the crawler.
2. At this point, we are investing our efforts in solving the problem of finding directory pages, we do not have certainty regarding the biography pages. Will the same URL classification techniques work?
3. Although we are reaching a very good Precision from the URL classifier, there are some false negatives in our testing results. The list below shows those misclassified URLs.
   - https://www.ecs.baylor.edu/index.php?id=961249 42 [0]
   - https://chemistry.rice.edu/chemical-biology 93 [0]
   - https://www.eiu.edu/math/personnel.php 132 [0]
   - http://www.chembio.t.u-tokyo.ac.jp/e/member.html 157 [0]
   - http://www.iitkgp.ac.in/department/CS/faculties 178 [0]
   - https://bioengineering.illinois.edu/directory/index.html 203 [0]

- https://www.baylor.edu/math/index.php?id=53580 215 [0]
- https://campusdirectory.morainevalley.edu/#/results-employees/MTC/department 221 [0]