

CS-410 Text Information System

MCS-DS Program (Fall 2021)

Course Project

[System Extension: ExpertSearch](#)

Team: JUEE

Snehangshu Shankar Bhattacharjee (ssb8)

Indranil Guha (iguha4)

Project Overview

ExpertSearch is a system which is very useful to search university websites for faculty information, biography, and key research areas. In this project, auto crawler feature is added to the existing system which enables automatic web scraping functionality, also topic mining for key research areas. Both features were implemented as a standalone python code.

Environment setup

This project developed and tested on Mac OS Big Sur. Due to python compatibility issues with Big Sur, Anaconda 2.0.4 was installed and then python 2.7 and python 3.5.6 runtime environments created. All the required python packages need to be installed in conda environment.

- gunicorn 19.10.0
- flask 1.1.4,
- requests 2.26.0
- bs4 0.0.1
- pytoml 0.1.21
- metapy 0.2.13
- genism 3.8.3
- sklearn 0.0
- scikit-learn 0.20.4
- numpy 1.16.1
- scipy 1.2.3

Create local repository using following command on a terminal to clone the github repository:

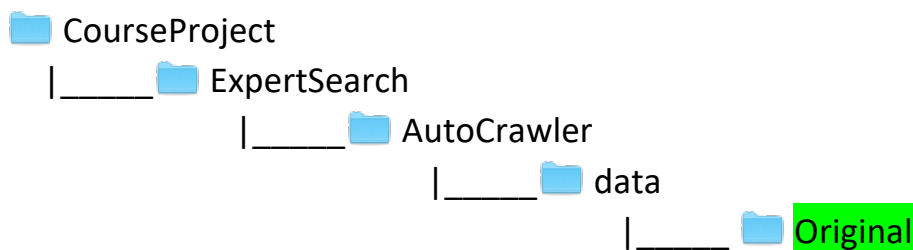
```
$ git clone https://github.com/snehangshugithub/CourseProject.git
```

Feature 1: Auto Crawler

Data file setup

Before running this utility, we need to put input data set in correct location. In our case we took university webpages from MP 2.1 data set as positive cases. We have also created negative test sets which consist of random websites. Both positive(known) and negative data set are kept in the below location.

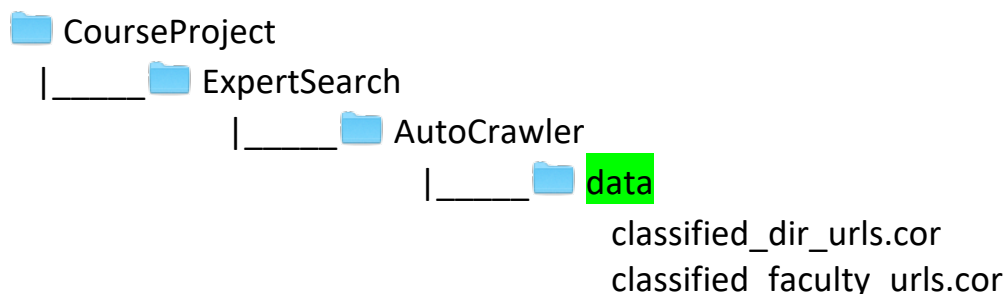
Input Files:



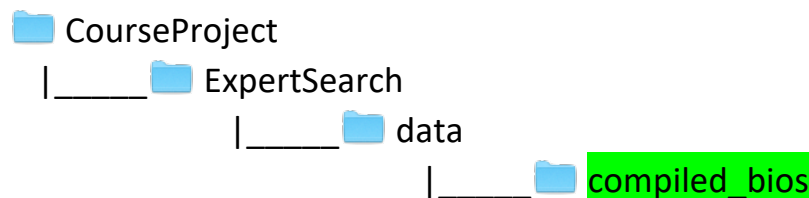
⇒ Known data set:
Faculty_Directory_Train.csv
Faculty_Directory_Test.csv
⇒ Negative data set:
Negative_case_set_1.cor
Negative_case_set_2.cor
Negative_case_set_3.cor
Negative_case_set_4.cor

Output Files:

Below files will be generated as output in the following location – classified directory and faculty url corpus.



At the end of the auto crawling process, bio information will be generated as text files for each faculty. In our case have 15520 text files generated in the location below.



Implementation

Auto Crawling feature is implemented as standalone python program where ***autocrawler.py*** is the main executable file, it is calling other sub-programs sequentially to perform different tasks and generating bios data for each faculty as final output.

The first task is to prepare the data set. The data preparation is handled by ***processor.py*** program by accessing input folder structure where both known faculty websites (positive case) and random websites (for negative case sample) URLs are kept. Then it mixes these 2 sources and partitions them into test and train URLs and finally prepare training and testing corpus files. These corpus files used as input for classification.

Text classification is performed by ***classifier.py*** program. This program takes train and test data set as input and label them appropriately. After reading the corpus of training data it builds a Doc2Vec model for feature vector representation of each document, then build the vocabulary and train the model first and then save the model to repurpose it for the same dataset for future run. Logistic Regression is used as a Discriminative Text Classifier to predict the categories of the test URLs given the test dataset.

crawler.py is used to facilitate to scrape faculty URLs. Traditional web-scraping technique used with BeautifulSoup for some clean-up tasks such as substituting newlines, tabs, whitespace removal and handle “403 Forbidden”, “404 Not found” error etc. Finally, it crawls classified faculty URLs and saves the contents of each bio

URL to a new file under ExpertSearch/data/compiled_bios. This bios data is further used to infer topic modeling.

constants.py is also used as a helper code to maintain references for all input and output folders and data files.

Code Execution (macOS)

Below is the step-by-step instruction to run this program.

1. Environment set up is the pre-requisites, conda environment for python 2.7 has been used.
2. Prepare input data set as mentioned in 'Data file setup' section.
3. Navigate to Auto Crawler directory

 CourseProject

| _____  ExpertSearch


| _____  AutoCrawler

 autocrawler.py

 classifier.py

 constants.py

 crawler.py

 processor.py

4. Run this command from terminal

\$ python autocrawler.py

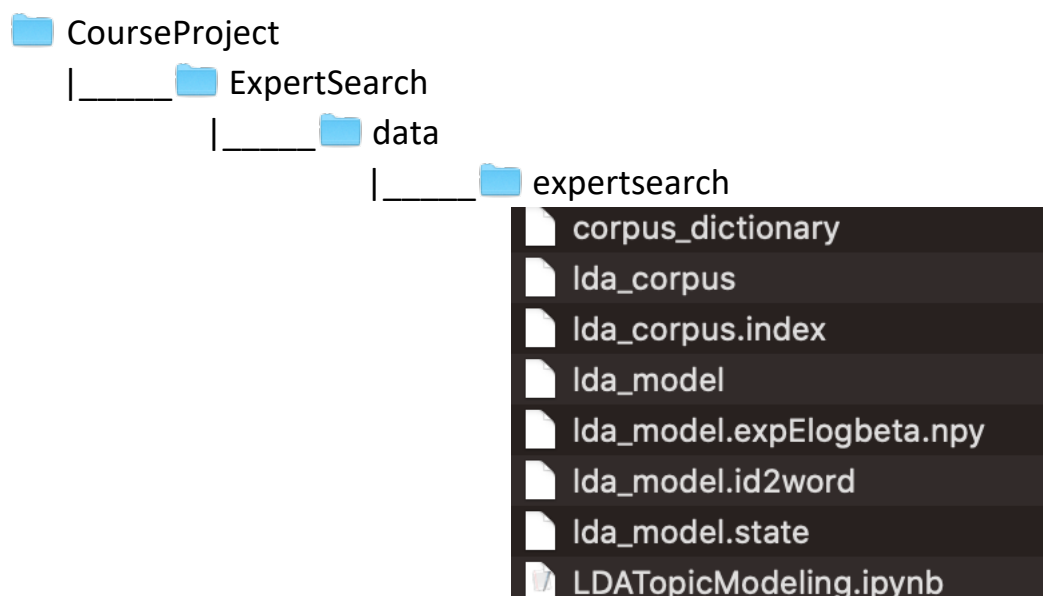
Note: Above step has been executed (took ~ 30 hrs. to complete) and bios data for each faculty kept in output folder 'compiled_bios' which can be used for topic mining (Feature 2).

Feature 2: Topic Mining

As part of this feature, we have explored an approach to create a topic model using genism LDA algorithm taking the bios data extracted through auto crawler as our text corpus. Coherence testing has been executed by varying the no of topics to obtain an optimal model that makes most sense in this context. Refer to the jupyter notebook file ***LDATopicModeling.ipynb*** for the steps in details.

In this approach, model creation required trial and error based manual efforts to get satisfactory term cluster, hence further tuning may be possible by tweaking the corpus such as creating filters for unwanted words based on the observation. Additionally, extracting and adding the faculty member research Interest pages to the corpus would improve it.

In this approach, we have saved the `lda_model` and `lda_corpus` in the below output path which can be further used to enhance the ExpertSearch system to add UI features such as showing user the topic word cloud by a query term. Also, we can use the model to show the user the list of top k topic terms as the research area by querying a faculty or faculty member.



Due to time and resource unavailability during this course work we couldn't finish the UI enhancement part, hence descoped it.

Task break up and contribution

Task description	Contributor	Efforts(hrs.)
Understanding ExpertSearch baseline code, troubleshooting issues with gunicorn installation in python environments, installing required python packages on Anaconda virtual environment.	Snehangshu Bhattacharjee, Indranil Guha	5+
Auto-crawling faculty pages to generate bio corpus for topic model inference	Snehangshu Bhattacharjee, Indranil Guha	20+
Topic mining on bio corpus using gensim LDA algorithm on jupyter notebook	Snehangshu Bhattacharjee, Indranil Guha	10+
Testing and Documentation	Snehangshu Bhattacharjee, Indranil Guha	5+
		40+