Project Progress Report 1

Abstract:

Using Data Mining techniques, our goal is to visualize the mention of “Chicago School” in the past literature and also learn a supervised classification algorithm to disambiguate each mention of “Chicago School” into one of the 40 labels (type of Chicago School).

Data:

Data Source:

1. Collection of all the Wikifier results of about 100,000 books. Each book is represented as a JSON file.
2. Collection of text snippets where “Chicago School” occurred in the book. Each text snippet is of length 42, where in “Chicago School” is in the middle and 20 words before and after are the exact 20 words before and after in the book.

Since all the books are not publicly available, currently we have only about 10 JSON files. Each JSON file is of size 1.5MB on disk. And, we only have few sample snippets, as the snippets extraction from the actuals books are not done yet.

Data Format:

Wikification is the task of identifying and linking expressions in text to their referent Wikipedia pages. The wikifier also identifies the name entities referred in the book.

The wikifier results on a book is a JSON object for the corresponding book (format is shown below).

{"id":"mdp.39015064582888","pages":[{"pid":"00000001","wikifier":[…],"ner":[…]} , …]}

- “id” is the book id.
- “pages” is the list of all pages in the book. Each page is uniquely mentioned by a page id (pid) contains the wikifier and NER tags in the page.
- “wikifier” is a list of all expressions with some metadata about their occurrence in the page of the book.
- NER is the list of all name entities referred in the page and few metadata about the occurrence.

An example of an item in wikifier is shown below:

{"text":"COLLEGE","link":"http://en.wikipedia.org/wiki/College","start":0,"end":7}
“text” is the entity of interest, “link” is the Wikipedia link for that expression, “start” and “end” are the start and end indices of the entity’s occurrence in the page.

An example of an item in NER list is shown below:

{"text":"VAN DYKE","type":"PER","start":43,"end":51}
“text” is a named entiry, “type” is any of PER/LOC/ORG/MISC, “start” and “end” are the start and end indices of the entity’s occurrence in the page.

An example of a text snippet:

“in the galleries and class-room to those who have applied. But no systematic effort has been made to introduce all Chicago school children to the Art Institute and to provide for them suitable instruction about the collections. To devise the best means”

Programming Languages and Packages:

Language: Python
Packages: nltk, sklearn, numpy, pandas, json, csv, os, sys, pickle, ngrams, re, collections

Preliminary Experiments:

I parsed the JSON files to extract the text and Wikipedia link in it, which we will be using in our later stages. Below is the code snippet:

```
import json, csv, os.path, sys
from os import listdir
links_file_path = "links.csv" //name of the output file
JSON_DIR = "ADM" //all json files are placed in this folder
file_list = [f for f in listdir(JSON_DIR)]
for file in file_list:
   with open("ADM/"+file) as data_file:
      d = json.load(data_file)
      book_id = d['id']
      pages = d['pages']
      links = []
      for i, item in enumerate(pages):
         wiki = item['wikifier']
         if wiki == []:
            continue
         else:
            for j, field in enumerate(wiki):
               data = field['link']
               if data == None:
                  links.append("Chicago_school")
               else:
                  links.append(data)
      mode = "w+"
      if os.path.isfile(links_file_path):
         mode = "a"
      with open(links_file_path,mode) as links_csv_file:
         try:
            write_handle = csv.writer(links_csv_file,dialect='excel')
            for item in links:
               write_handle.writerow([book_id,item])
         except csv.error as e:
            sys.exit('file %s, line %d: %s' % (links_file_path, write_hanle.line_num, e))
```

Different tasks in the project:

1) Compute a graph for each JSON file, where all NER tags are nodes and there is a link from node A to node B if B occurs after A. Compute the node weight (number of incoming links) of all nodes. This helps us to identify the frequent topics in the book.
2) Run n-grams over Wikipedia links in JSON files or the CSV file (created above).
3) Train a classifier to classify the JSON files to one of the 40 labels (type of Chicago School) using supervised learning.
4) Train a classifier to classify the text snippets to one of the 40 labels (type of Chicago School) using supervised learning.
5) Compute metrics to assess accuracy of wikification (JSON input files), perform steps on cleaning systematic errors.
6) Compute metrics to assess final classification accuracies (3, 4)

Task that I am responsible for:

Train a classifier to classify the text snippets to one of the 40 labels (type of Chicago School) using supervised learning. (Tasks 4 in the above list)

- I have preprocessed the data (few sample snippets): removed English stop words, punctuations, stemming using below functions from nltk package:
  stopwords.words("english"), porter_stemmer.stem(word), string.punctuation
- I am thinking of trying RandomForest, Support Vector Machine and Logistic Regression for the classification and I am currently studying these classifiers.