Information Retrieval and Extraction

Implementation for An Automatic Course

Website Discovery

INFSCI 2910-Independent Study

Zexin Zhao
University of Pittsburgh
ZEZ7@pitt.edu

1. INTRODUCTION

Sharing is a goal of the Internet development. As more and more educational resources are open online, taking this advantage of online resources to enrich the content and relevant materials of University courses is a good choice for educator to improve the course quality. However, because online scattered information contains many overlapping contents and is organized in diverse forms, integrating them is a challenge.

This research aim to gain two goals: 1) Modeling online course; 2) Modeling courses by mapping modules into uniform representation. In order to do an efficient work for accomplishing these two tasks. This project will implement some tools to complete information retrieval and extraction.

2. MODELING ONLINE COURSE

In this task, the main goal is to store and present online courses with a uniform template, including organizing textbooks, slides, and readings. Classifier is needed to identify relative course website. All the data that should be fed into the classifier is collected by a web crawler and a crowdsourcing system.

2.1 Preparing Raw Data

On this stage, we need to collect relevant online educational resources for specific courses no matter whether it is useful or not.

2.1.1 Required Features of Data

For each relevant online educational resource, we need an information collection including link, title, an ID, and the course's category it belongs to. In order to organize those links by the course's categories, we use search engine to select relevant links and other information for us according to the course's categories.

2.2 Collecting Training Set

On this stage, we need to collect training data set for the classifier. The data in training data set should be done the "valid-identification" by specialists.

2.2.1 About "Valid-identification"

If an online educational resource is valid, it should have these requirements showing below:

- Relevant to the required course's category;
- Having a valid link;
- There must be a schedule or a syllabus on the page which the valid link lead to;
- The schedule or a syllabus should provide links of the materials it refers to. This task requires specialists to identify valid resources in raw data.

2.2.2 About Specialists

Specialists in our project could be educator, students or other people who have basic educational background and an ability of identifying valid resources.

2.3 Implementation

In order to accomplish these two steps efficiently, I implemented applications for both two steps.

2.3.1 Web Crawler

I used Python and its library: Beautiful Soup to implement this crawler. The target of crawler is Google search. The complete structure of the crawler shows in Figure 1.

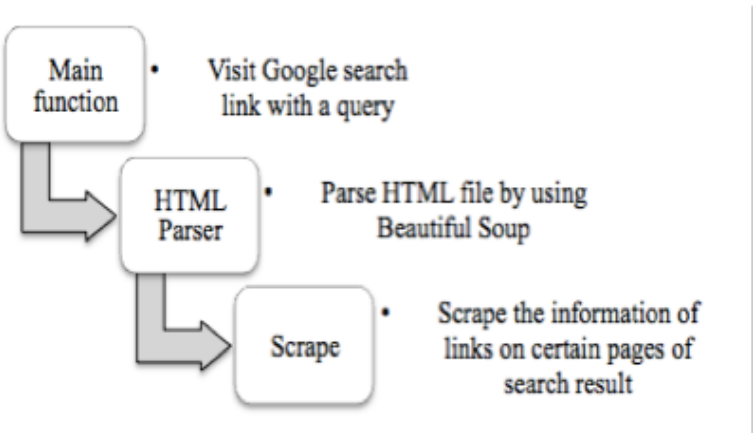

Figure 1. The structure of the crawler

We have four course's categories: Information Retrieval, Human Computer Interaction, Database, and Java Programming. For each course's category, I configure the amount of pages the crawler will work on is 10. The outputs are stored in files with JSON format. In order to cover as many courses as possible, I also use different course names as queries for one category. One part of crawling result shows in Figure 2.


```
{
"links":[
{"link": "http://www.uio.no/studier/emner/matnat/ifi/INF9260/", "ID": 130, "name": "INF9260 -Human-computer
interaction(HCI) - University of Oslo"},
{"link": "http://edu.mah.se/en/Program/TAINE", "ID": 29, "name": "Interaction Design, Master&#39;s Programme
(One-Year ... - Malm\u00f6"},
{"link": "http://www.philau.edu/msintdesign/", "ID": 123, "name": "MS in User Experience andInteraction Design-
Philadelphia ..."},
{"link": "http://www.cs.umd.edu/hcil/academics/courses.shtml", "ID": 30, "name": "HCI Coursesat UMD - UMD
Department of Computer Science"},
{"link": "https://www.ntnu.edu/studies/courses/IT8002", "ID": 90, "name": "Course- Advanced Topics
inHuman-Computer Interaction- IT8002 ..."},
{"link": "http://www.dundee.ac.uk/study/ug/applied-computing-human-computer-interaction/", "ID": 51, "name":
"Applied Computing:Human Computer InteractionBSc ..."},
{"link": "https://onderwijsaanbod.kuleuven.be/syllabi/v/e/G0Q55AE.htm", "ID": 133, "name": "Fundamentals
ofHuman-Computer Interaction- KU Leuven"},
{"link": "https://www.youtube.com/user/InteractionDesignOrg", "ID": 64, "name": "Interaction-Design.org -
YouTube"},
{"link": "http://www.hfg-gmuend.de/en/Information_about_study.html", "ID": 97, "name": "Interaction DesignBA -
Schw\u00e4bisch Gm\u00fcnd"},
{"link": "http://info.sjsu.edu/web-dbgen/catalog/courses/ISE217.html", "ID": 56, "name": "ISE 217:Human Computer
Interaction- Info.sjsu.edu"},
{"link": "https://www.pce.uw.edu/courses/interaction-design-and-prototyping-hcde-536", "ID": 59, "name":
"Interaction Design&amp;amp; Prototyping (HCDE 536) - UW ... - Seattle"},
{"link": "https://github.com/gzuidhof/ahci", "ID": 134, "name": "GitHub - gzuidhof/ahci: Advances inHuman Computer
Interaction..."},
{"link":
"http://www.elo.iastate.edu/graduate-degrees/human-computer-interaction-masters-of-science-degree-program-online/"
, "ID": 18, "name": "Human Computer InteractionMaster&#39;s of Science Degree Program ..."},
{"link":
"https://www.prospects.ac.uk/universities/university-of-hertfordshire-3783/courses/human-computer-interaction-onli
ne-41808", "ID": 29, "name": "Human Computer Interaction(Online) | Prospects.ac.uk"},
{"link": "http://mshci.gatech.edu/", "ID": 17, "name": "Georgia Tech Master&#39;s Program inHuman-Computer
```


Figure 2. One part of crawling result for HCI course

2.3.2 Link-mark System

In this part, I will explain the implementations in two phases.

1) Crowdsourcing on Amazon Mechanical Turk

For the second step, we need allow specialists to visit those links in crawler's output files, and mark it if it's a valid link. Besides, specialists are also required to input the location of a syllabus or a schedule on the page.

I implemented a RESTful web server to record specialists' visiting behavior and decisions. The workflow of the system shows in Figure 3.

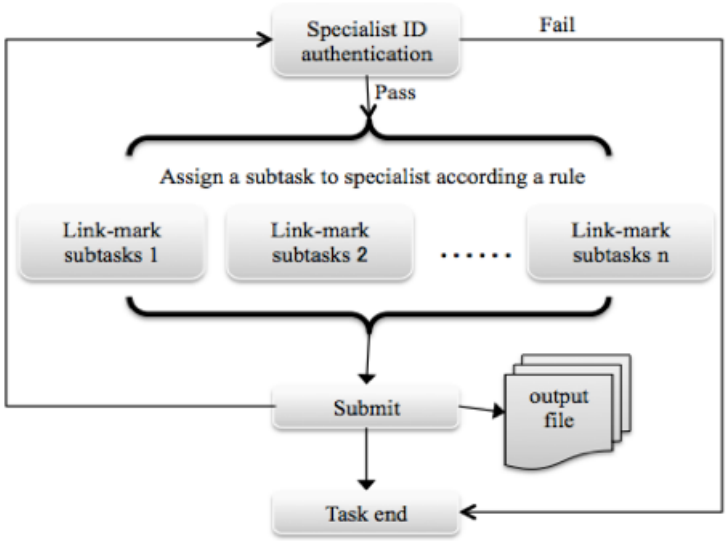

Figure 3. The workflow of the system

For each course's category, I configure the amount of links shows in a subtask is 5. The output files are stored with JSON format. The display of the system shows in Figure 4. One part of output file shows in Figure 5.

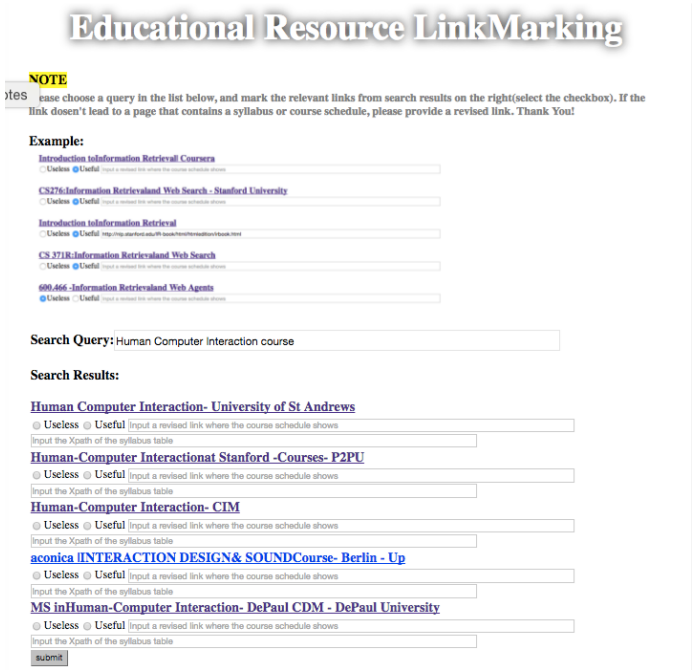

Figure 4. The display of the system

Figure 5. One part of output file

In order to collect large amount of diverse training data, we planed to use online crowdsourcing platform to deploy this system. Amazon Mechanical Turk (MTurk) is a crowdsourcing Internet Marketplace enabling individuals and businesses (known as Requesters) to coordinate the use of human intelligence to perform tasks that computers are currently unable to do. The final display of get-start page on MTurk shows in Figure 6.

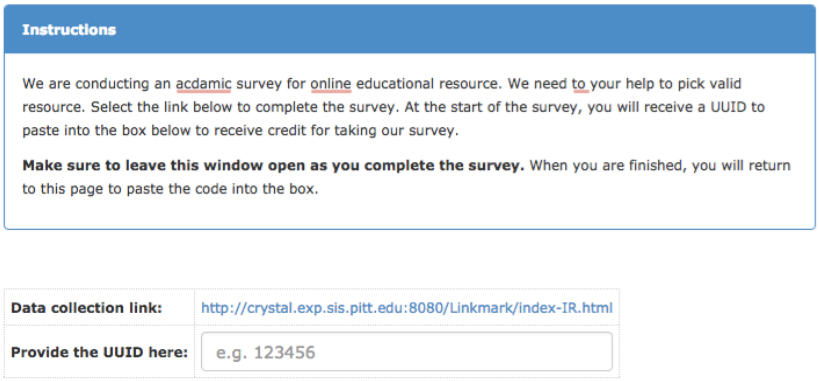

Figure 6. The final display of get-start page on MTurk

However, this system on MTurk didn't work very well and less efficient because the workflow that MTurk required is a little complicated. And we don't think using crowdsourcing can help the second step a lot. So we turn to deploy it on a local server.

2) Collecting training data on local server

The Link-mark system has no different with the system on MTurk. Now we are still working on import data in this system.

3. MODELING ONLINE COURSE

In this task, the main goal is to map course resources to chapters of textbook, integrate various resources. In order to complete this task, we need to extract course syllabus from course websites, and download materials like readings, and slices.

3.1 Syllabus Extractor

3.1.1 The design of Syllabus Extractor

According to collection of useful courseweb link, this information extractor should identify a valid syllabus in the webpage and restore its information into a file with a uniform representation. However, some of syllabuses are organized using table or list. They are easier to be recognized than other syllabuses that are organized without a specific format. So we chose these syllabuses as our target to extract.

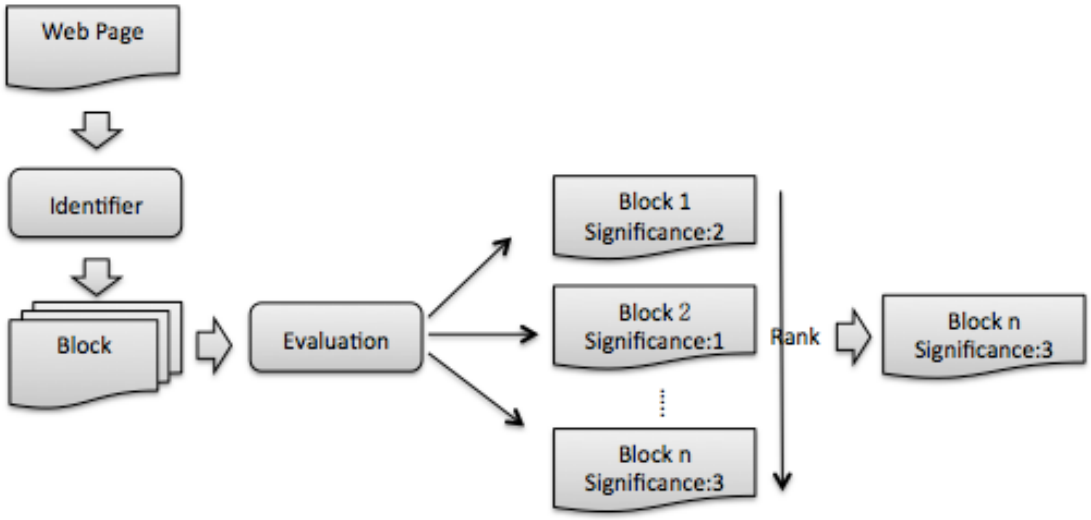

Figure 7. Syllabus Extractor's structure

Extractor's structure is shown above (Figure 7). The mechanism of this extractor is to pick every block in the webpage, and then evaluate its significance based on specific features that a syllabus should have, finally output the block with highest significance in each website. There are three main features that a syllabus "table" may have: A. the number of rows is more than N; B. content should contain positive words; C. the titles of table should contain keywords. There are three main features that a syllabus "list" may have: A. the number of rows is more than N; B. content should contain positive words. Notice that N, positive words list, and keywords list are generated according to the observation to as mush as online resources, Initially, I defined N=8, the positive words and keywords are ones that are highly used in syllabus of IR course or other specific

course. The evaluation process will calculate significance of a block based on a formula below (Formula 1). Then ranking all the blocks and output the blocks with highest significance.

$$Significance=nrow(T)/N + t(kw)/ave(kw) + t(pw)/ave(pw)$$

Formula 1. nrow() is a function to get number of rows of a Table. T represents a table needed to be evaluated. t(kw) function is to count the number of keywords shows in this table title. ave(kw) is the average amount of keywords shows in syllabuses that I observed. t(pw) function is to count the number of positive words shows in this table title. ave(kw) is the average amount of positive words shows in syllabuses that I observed.

3.1.2 Implementation

This tools is used BeautifulSoup library in Python to parse an html website and help to extract every blocks from the website. The result is shown as following(Figure 8). The precession is 93% to all tables and lists. Those missing tables and lists can't be recognized because they are organized with mess format. Therefore, they need to be extracted manually.

the syllabus in http://www.cc.gatech.edu/~stasko/6750/syllabus.html

table

The majority of readings listed below are chapters or sections from our textbook, *Human-Computer Interaction*, 3rd edition, by Dix, Finlay, Abowd and Beale, Prentice Hall, 2003. As you can see, we will be skipping around in terms of reading assignments, so keep up and listen in class for any changes or modifications. We also will use the book *The Design of Everyday Things* by Don Norman as a second course text. A number of supplementary readings will be passed out in class as well. There is *no excuse* for ignorance of the assigned reading material.

Date	Topic	Reading	Lecture	Assignments
Week 1				
Jan 9	Introduction & History of HCI	Intro	PPT, B/W	
Jan 11	Project overview, IRB, UCD		PPT, B/W	HW 1, P0
Week 2				
Jan 16	Usability principles	2,3,4,7	PPT, B/W	
Jan 18	Human abilities	1	PPT, B/W	P1
Week 3				
Jan 23	Predictive Evaluation	9	PPT, B/W	
Jan 25	Understanding Users, Reqts. Gathering		PPT, B/W	
Week 4				
Jan 30	Task analysis	13,15	PPT, B/W	
Feb 1	DOET	DOET	PPT, B/W Abowd PPT	
Week 5				
Feb 6	Design	5	PPT, B/W	
Feb 8	Graphic design		PPT, B/W	HW 2, P2

Figure 8.

3.2 Course Materials Download and Parser

The research hopes to map all materials (readings, slides) to chapters, so we next pick up those syllabuses that mentioned chapter information. This work was done manually, and got a .xml file as follows (Figure 9).

```
<coursename name="Information Retrieval">

  <course no="001">
    <url>http://web.stanford.edu/class/cs276/#syllabus</url>
    <textbook>IIR</textbook>
    <textbook>MIR</textbook>
    <units>
      <unit>
        <no>1</no>
        <title>Introduction to the course</title>
        <chpter>IIR Ch. 1</chpter>
        <slide_url>http://web.stanford.edu/class/cs276/handouts/lecture1-intro.ppt</slide_url>
        <reading>
          <name>IIR Ch. 1</name>
          <reading_url>http://nlp.stanford.edu/IR-book/pdf/01bool.pdf</reading_url>
        </reading>
      </unit>
      <unit>
        <no>2</no>
        <title>Merge algorithm for proximity queries using a positional index</title>
        <chpter>IIR Ch. 2</chpter>
        <slide_url>http://web.stanford.edu/class/cs276/handouts/lecture2-dictionary.ppt</slide_url>
        <reading>
          <name>Porter's stemmer (MIR)</name>
          <reading_url>http://www.sims.berkeley.edu/~hearst/irbook/porter.html</reading_url>
        </reading>
        <reading>
          <name>Porter stemming algorithm (Official) </name>
          <reading_url>http://www.tartarus.org/~martin/PorterStemmer/</reading_url>
        </reading>
        <reading>
          <name>A skip list cookbook (Pugh 1990) </name>
          <reading_url>http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.17.524</reading_url>
        </reading>
        <reading>
          <name>Fast phrase querying with combined indexes (Williams, Zobel, Bahle 2004)</name>
          <reading_url>http://portal.acm.org/citation.cfm?id=1028102</reading_url>
        </reading>
        <reading>
          <name>Efficient phrase querying with an auxiliary index (Bahle, Williams, Zobel 2002)</name>
```

Figure 9. syllabus with chapter information

Next step, I need to download every material listed in this file, and store them in disk. Finally, I parsed every material and got a bag of words for each file. The workflow of this tool is shown below (Figure 10).

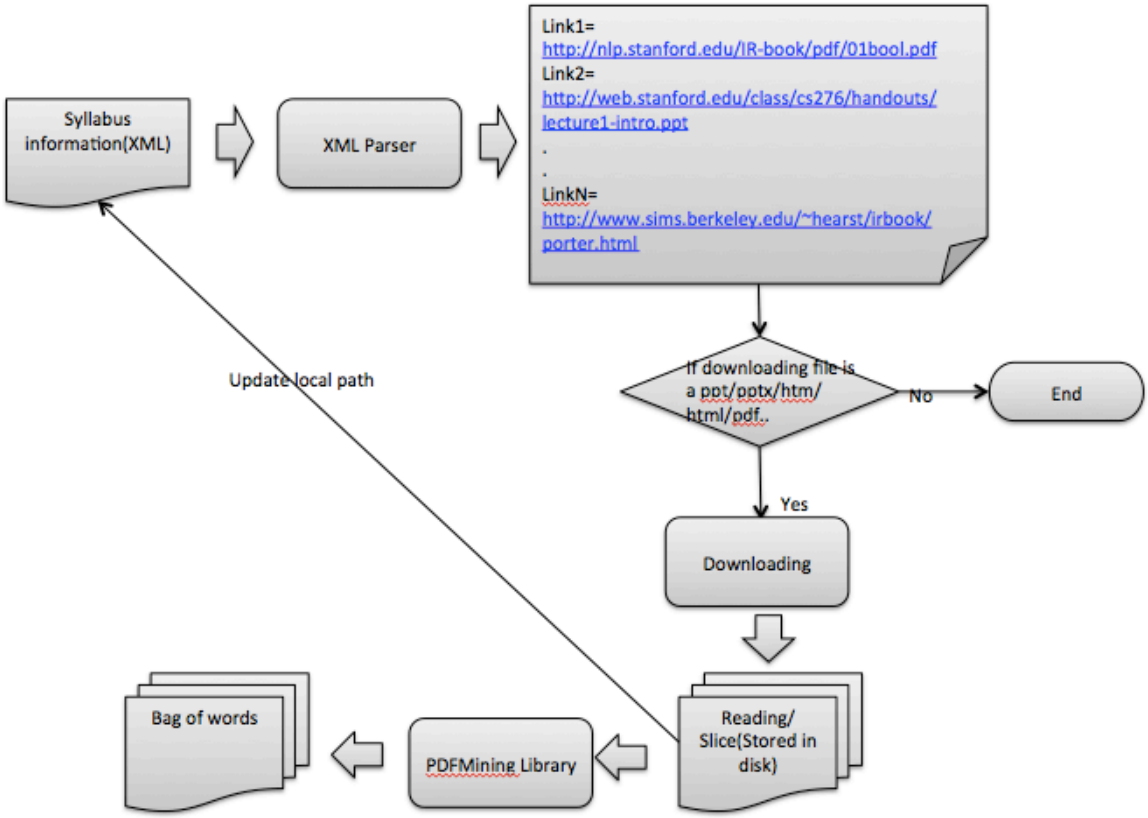

Figure 10. The workflow of material downloading and parser.

3.3 Wiki Article Extractor and Concept Picker

In order to map materials, we need use concepts from Wikipedia in specific field. I implement an article extractor to search and store all articles belongs to "Computer Science" field. Those article are from dump file (XML) of Wikipedia. The rule of article being selected is that the article should contain category information, which belongs to "Computer Science" category and its subcategories. Category links of Wikipedia are stored in MySQL database, because Wikipedia only provide whole category tree as SQL file. Therefore, the extractor works as follows (Figure 11).

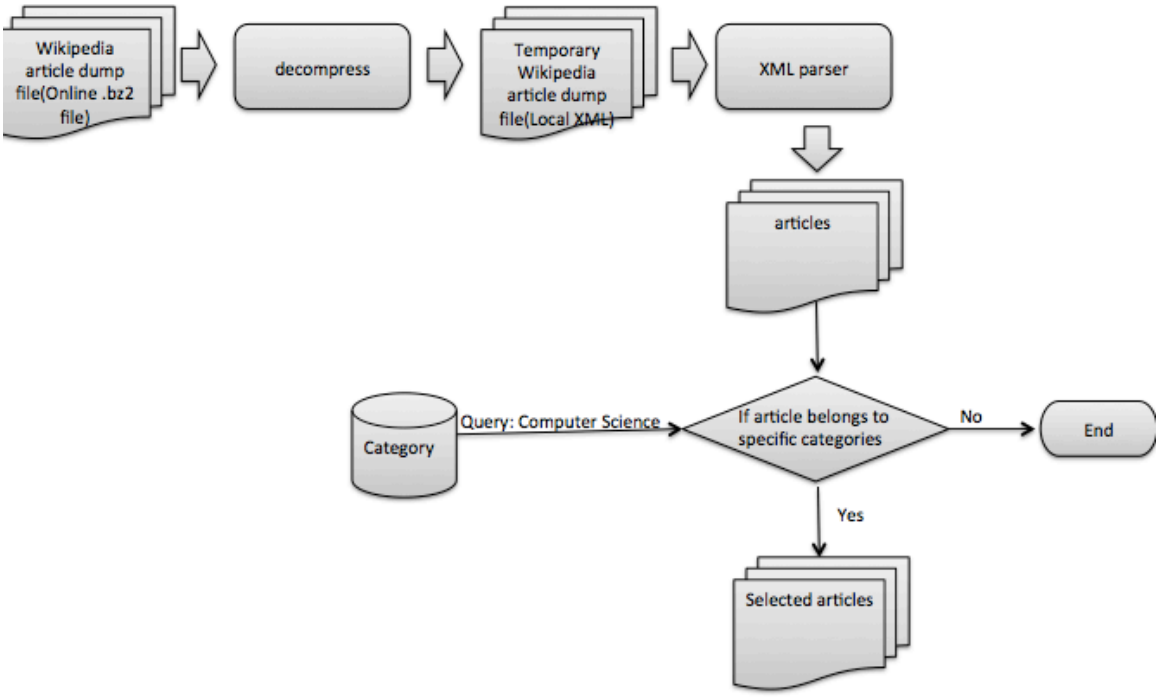

Figure 11. The workflow of Wikipedia article extractor

The next step, we need to pick concepts referred in articles. This tool mostly uses regular expression to identify referred concepts marked using specific pattern. In dump files, the referred concepts are in "[[ ]]". So the regular expression is ***r''\[\[.+?\]\]''***. Besides, before extracting concepts, these articles should be removed ads and unrelated sections like "navigation menu", "External links", "Further reading", and so on.
By now, this tool can select the concepts. However, because we have very large amount of wanted articles, parsing one file takes long time. So I'm still working on this tool to improve its online processing ability by using cache technique in Python.

4. CONCLUSIONS AND FUTURE WORK

Most of the implementation in this project has accomplished. The following stage will be using the relationship between materials' content and textbook, and the concepts from Wikipedia to learn a connection between course materials and units that a course schedule contains.

5. ACKNOWLEDGEMENT

I wish to thanks the following people for their contribution to this project:

Prof. Peter Brusilovsky and Daqing He who gave us the knowledge resource and opportunity to design and develop this project. Also they provided useful suggestions and comments during the development process. Besides, Rui Meng and Shuguang Han who gave a great help in problems solving.