Vertical Search for Courses of UIUC

http://greedy.cs.uiuc.edu/dssi/course/search.php

Jessica Bell, Alexander Loeb, Sharon Paradesi, Michael Paul, Jing Xia, Jie Zhang

The aim of the Course Search project is to construct a database of UIUC courses across all departments ultimately creating a centralized knowledgebase about each course. This knowledgebase is to then be augmented by drawing relations between courses both within and between departments and further by finding similarities among courses outside of the University of Illinois. These relations will be formed via varied machine learning and clustering techniques where course similarity is measured by course content itself. The content of each course will be determined by extracting information about each course from a number of sources.

Data Extraction

Course Catalog

The course catalog and Fall 2008 schedule is available in PDF format with every course schedule within a department listed in a single document. We converted these department catalogs into HTML files which could then be parsed as text. A PHP script was written to parse the HTML files using regular expressions.

Text Book

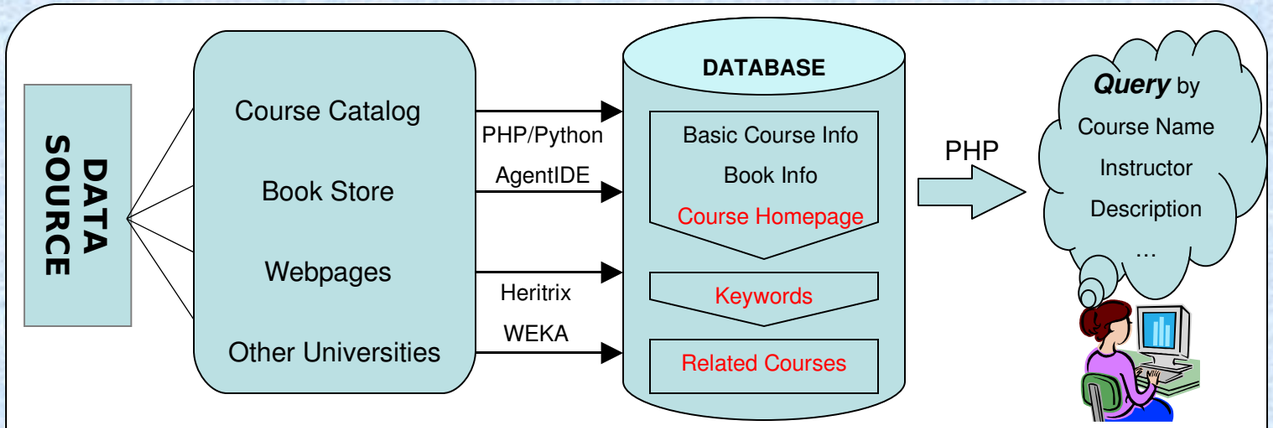
Course textbooks were obtained both by extraction from course homepages where possible as well as by a script written to manipulate the back end javascript calls embedded within the Illini Union Bookstore website.

Webpages

We used Heritrix, an open-source crawling software package, to perform crawling on the domain of computer science at UIUC. In our experiment, we tuned the parameters to filter out non-html pages (e.g., pdf files, jpeg files, etc.). Report shows that most of CS websites under UIUC domain were covered.

Additional

The UIUC people directory was employed for additional information based on a search of instructor names.



Homepage Classification

To classify a webpage is course homepage or not, we used some CS course homepages as **training data** labeled as positive examples. False training data were randomly chosen from CS webpages which were not homepages.

The list of **features** used are – course, semester, homework, textbook, syllabus, instructor, etc.. A feature can consist of multiple keywords. Each html file was parsed and represented as a feature vector. Each dimension of the vector represents how many times the feature appears in that page.

We used WEKA for the **classification process**. The script was run for the training set first. Accuracy was tested by running the classifier on a test data set and achieved an accuracy of 89.5% so far.

Keywords

First, calculate the informativeness of words using Gain Ratio measure and set threshold to find “common words”. **Gain Ratio** is a normalized variant of **Information Gain** that is less biased toward frequent terms.

Then split course descriptions by commas, semicolons, and common words. Score remaining phrases by summing their Gain Ratio measures. Also pull 1- and 2-grams from these phrases and assign a fraction of the original score.

Related Courses

Link courses by weighted edges whose weights are calculated by summing the scores of the courses’ matching keywords.

Interface

The front-end of the project uses PHP. The user can search by course name, instructor name, or course description. The search script queries the database by looking for matches with words in the user's query. Results are scored and ranked by term frequency, with scores being boosted by other features such as exact phrase matching and if the term appears in the course name. The courseID of a search result is passed in through the URL when the user clicks the name, and the PHP web pages to display the course information query the database using the supplied courseID.

Example

Query: “shakespeare”

Returns:

ENGL 218: Introduction to Shakespeare
CWL 241: Lit Europe & the Americas I
CLCV 120: The Classical Tradition
THEA 474: Acting Studio III: Acting

Similar to ENGL 218:

ENGL 113: Intro to Comedy
ENGL 418: Shakespeare I
ENGL 209: English Lit to 1798