Focused Retrieval of University Course Descriptions from Highly Variable Sources

Thomas Effland - SUNY University at Buffalo

Motivating Question
How can we automatically retrieve semantically similar content (such as university course descriptions) from many disparate sources on the Web that do not reference each other when we only know the domain names and have limited computational and training resources?

Challenges
- Target content is typically very sparse on large sites, so brute force crawling is unreasonable.
- Organizational structure and content location vary highly for each site, thus canonical rule-based approaches are ineffective.
- Typical topical-locality [1] assumptions made in focused web crawling do not hold when sites do not reference each other.
- Retrieving relevant content requires identifying and tunneling through irrelevant pages [2] that lead to target content.
- Gathering hand-labeled data is costly.

Webpage Representation
Each page is represented by a feature vector of the page content and a set of feature vectors for each link on the page.

Page Features
- TF-IDF [3] of words and bigrams of segment url
- TF-IDF of words and bigrams of the title
- Latent Semantic Analysis (LSA) [4] of TF-IDF of words and bigrams of page body words

Url Features
- TF-IDF of words and bigrams of the link anchor text

Defining a Relevance Metric
To obtain a measure of how close an irrelevant page is to a target content page, we define the label \( R \) of a page as the normalized link distance from the page to the target.

\[
R(P) = 1 - \frac{\text{LinkDist}(P, T)}{\text{LinkDist}(S, T)}
\]

This relevance metric helps address variable structure of sites.

Training Stage
1) User marks sample traversal paths using a simple Chrome extension called MarkIt.
2) Labeled and unlabeled data are collected from paths and training data is extracted.
3) Two Random Forest Regressors [5] are fit to training data.
4) Regressors are used to generate more training data from unlabeled data by ranking pages and labeling highly ranked pages or asking user for input on middle ranked pages. This semi-supervised approach combines self-training [6] with active learning [7] and saves considerable time in generating large training set from significantly less intervention.

This yields flexible regressors from only a small number of sample traversals from a few sites.

Deployment Stage
1) Top level url is input in queue. Queue pops top url.
2) Features for each url on page are extracted and urls are ranked by relevance prediction, then pushed into priority queue.
3) Features for page content are extracted and relevance is predicted. If relevance prediction for the page is high and agrees within threshold amount with initial prediction from its url, the page is classified as a target page. The user may review the proposed retrieval and classify the prediction as correct or not and the page is added to the training data. This is an example of active learning.
4) The next page from the queue is popped and the cycle continues until user-specified endpoint.

Algorithm & System Design

References