Exploiting Web Search to Access IEEE papers

Bidisha Roy
Associate Professor
St. Francis Institute of Technology
Mumbai, India

Joy Machado
BE (Computer Engineeing)
St. Francis Institute of Technology
Mumbai, India

Melcia Raj
BE (Computer Engineeing)
St. Francis Institute of Technology
Mumbai, India

Gnana Sonica Nadar
BE (Computer Engineeing)
St. Francis Institute of Technology
Mumbai, India

ABSTRACT

Web People Search (WePS) is a desktop application which aims to find relevant pages from the web related to a person's name. The project work is targeted to design an advanced version of the deep extraction tool using Clustering Algorithm. In this research work, the focus is mainly on querying for personal information of scientists and researchers. The user has to set the proper target name for search, which when completed, the user will receive complete PDF files based on the search. Each group of information items (cluster) will be defined by its key and the user make the choice. The result page will be produced from the chosen clusters. For making the search operationally accurate, we will assume the usage of research and conference doc files as they carry a standard format of name, e-mail ID, publication, images, and links to the full images.

General Terms

API (Web service platform), PDF files, Java Swing, Net beans.

Keywords

Clustering, Web People Search, WePS, Named Entity, Web Querying, Crawling, Indexing.

1. INTRODUCTION

World Wide Web has more and more online Web databases which can be searched through their Web query interfaces. The number of Web databases has reached 25millions according to a recent survey. All the Web databases make up the deep Web (hidden Web or invisible Web). Often the retrieved information (query results) is enwrapped in Web pages in the form of data records. These special Web pages are generated dynamically and are hard to index by traditional crawler based search engines, such as Google and Yahoo. Each data record on the deep Web pages corresponds to an object. In order to ease the consumption by human users, most Web databases display data records and data items regularly on Web browsers.

Searching for people on the Web is one of the most common query types to the web search engines today. Internet users access billions of web pages online using search engines. However, when a person name is queried, the returned result often contains web pages related to several distinct namesakes who have the queried name. The task of disambiguating and finding the web pages related to the specific person of interest is left to the user. The approach is based on extracting named entities from the web pages and then querying the web to collecting co-occurrence statistics, which are used as additional similarity measures.

One particular case of this people-document association task is referred to as personal name resolution. The task is as follows: given a set of documents all of which refer to a particular person name but not necessarily a single individual (usually called referent), identify which documents are associated with each referent by that name. Different methods have been used to represent documents that mention a candidate, including snippets, text around the person name, entire documents, extracted phrases, etc.

1.1 Existing System and its effect

Searching for information on the Web is not an easy task. Searching for personal information is sometimes even more complicated. Below are several common problems we face when trying to get personal details from the web:

- Majority of the Information is distributed between different sites.
- It is not updated.
- Multi-Referent ambiguity – two or more people with the same name.
- Multi-morphic ambiguity which is because one name may be referred to in different forms.
- In the most popular search engine Google, one can set the target name and based on the extremely limited facilities to narrow down the search, still the user has 100% feasibility of receiving irrelevant information in the output search hits. Not only this, the user has to

manually see, open, and then download their respective file which is extremely time consuming. The major reason behind this is that there is no uniform format for personal information.

Maximum of the past work is based on exploiting the link structure of the pages on the web, with hypothesis that web pages belonging to the same person are more likely to be linked together.

1.2 Motivation

There is no known application/site that performance similar tasks. There are tools which use static internet-based databases for finding personal details. These tools are problematic because they are not up-to date and are limited. Google is a crawler based search engine, but again the process of search and download becomes manual when it comes to need to find all the research papers of only one author. This is definitely a challenging area where are more than 5000 MNC organizations are currently conducting research for advance search engine, but none of them has yet come out with commercial product usage like google.com, altavista.com, ask.com, thereby posing higher feasibility and worth of doing research and development in Academic level too.

1.3 Proposed System

One of the key challenges that needs to be overcome to make the project functionality a reality, is to build an advance query system that is capable of reaching high disambiguation quality. The system is targeted to design an advance version of the deep extraction tool using Clustering Algorithm. The visual information of Web pages can be obtained through the programming interface provided by Web browsers. A Visual Block tree is actually a segmentation of a Web page. The root block represents the whole page, and each block in the tree corresponds to a rectangular region on the Web page. The leaf blocks are the blocks that cannot be segmented further, and they represent the minimum semantic units, such as continuous texts or images. Fig. 1a shows a popular presentation structure of deep Web pages and Fig. 1b gives its corresponding Visual Block tree. An actual Visual Block tree of a deep Web page may contain hundreds even thousands of blocks. Fig. 1c represents the layout of deep web pages.

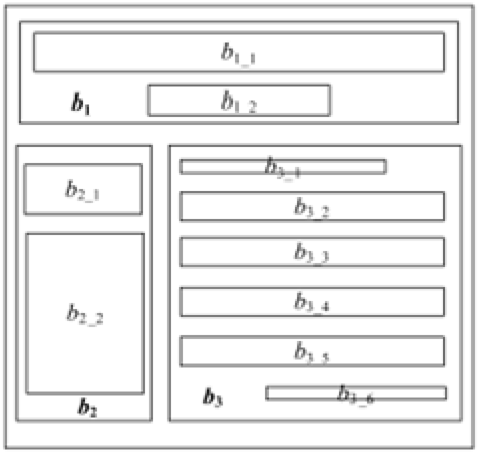

Fig. 1. (a) The presentation structure

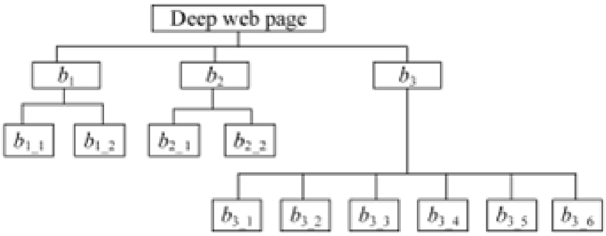

Fig. 1. (b) Visual Block tree

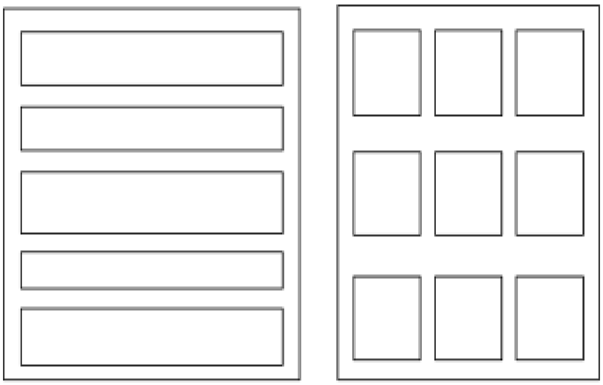

Fig. 1. (c) Layout models of data records on deep Web pages

2. SYSTEM OVERVIEW

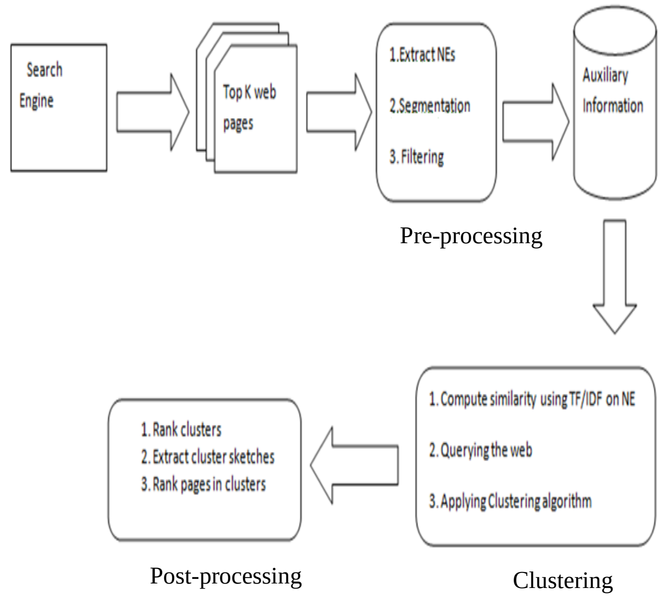

Fig 2: System Overview

The main steps in WePS are:

1. User Input: The input to WePS is a person's name via the user interface. The system sends a query consisting of a person name to a search engine, such as Yahoo!, and retrieves the top-K web pages.
2. Pre-processing: The main pre-processing steps are:
   - Extraction: Named Entities, specifically people are extracted.
   - Segmentation: If algorithm detects that a web page might refer to multiple name sakes, this web page is segmented into sub-web pages.
   - Filtering: Certain named entities are too ambiguous to be part of queries, so they need to be filtered.
3. Clustering: The filtered web pages are then clustered and saved.
   The clustering part consists of the following stages:
   - TF/IDF Similarity: TF/IDF similarity on NEs only is computed.
   - Querying the Web: For each pair of web pages di and dj several co-occurrence queries are formed and issued to a Web search engine.
   - Applying Clustering: A clustering algorithm, such as nearest neighbor algorithm are used and takes into account the TF/IDF similarity.

4. Post-processing. The post-processing steps include:
   a. Cluster Sketches are computed.
   b. Cluster Rank is computed based on
      i. the context keywords, if present and
      ii. the original search engine's ordering of the web pages.
   c. Webpage Rank is computed to determine the relative ordering of web pages inside each cluster.

User Result: The resulting clusters are presented to the user, which can be interactively explored.

3. FLOW CHART

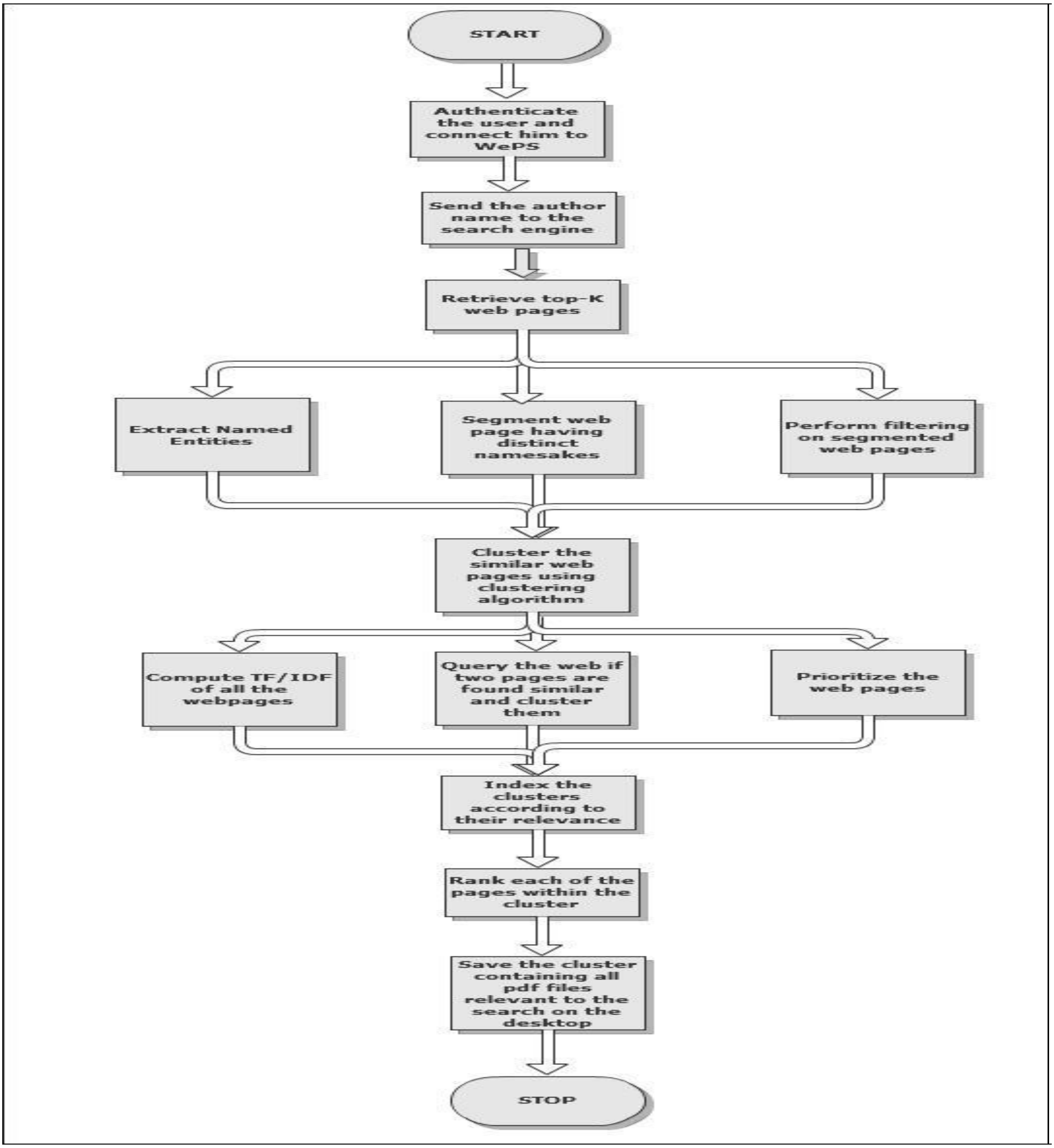
START
Authenticate the user and connect him to WePS
Send the author name to the search engine
Retrieve top-K web pages
Extract Named Entities
Segment web page having distinct namesakes
Perform filtering on segmented web pages
Cluster the similar web pages using clustering algorithm
Compute TF/IDF of all the webpages
Query the web if two pages are found similar and cluster them
Prioritize the web pages
Index the clusters according to their relevance
Rank each of the pages within the cluster
Save the cluster containing all pdf files relevant to the search on the desktop
STOP

Fig 3: Flow Chart

4. ALGORITHM AND WORKING

A. MINI-BATCH K- MEANS

The *k*-means optimization problem is to find the set *C* of cluster centers $\mathbf{c} \in \mathrm{R}^m$, with $|C| = k$, to minimize over a set *X* of examples $\mathbf{x} \in \mathrm{R}^m$ the following objective function:

$$\min \sum_{x \in X} ||f(C, \mathbf{x}) - \mathbf{x}||^2$$

Here, $f(C, \mathbf{x})$ returns the nearest cluster center $\mathbf{c} \in C$ to $\mathbf{x}$ using Euclidean distance. It is well known that although this problem is NP-hard in general, gradient descent methods converge to a local optimum when seeded with an initial set of *k* examples drawn uniformly at random from *X*.

```
Given: $k$, mini-batch size $b$, iterations $t$, data set $X$
Initialize each $\mathbf{c} \in C$ with an $\mathbf{x}$ picked randomly from $X$
 $\mathbf{v} \leftarrow 0$
 for $i = 1$ to $t$ do
     $M \leftarrow b$ examples picked randomly from $X$
     for $\mathbf{x} \in M$ do
         $\mathbf{d}[\mathbf{x}] \leftarrow f(C, \mathbf{x})$ // Cache the center nearest to $\mathbf{x}$
     end for
     for $\mathbf{x} \in M$ do
         $\mathbf{c} \leftarrow \mathbf{d}[\mathbf{x}]$ // Get cached center for this $\mathbf{x}$
         $\mathbf{v}[\mathbf{c}] \leftarrow \mathbf{v}[\mathbf{c}] + 1$ // Update per-center counts
         $\eta \leftarrow 1/\mathrm{v}[\mathrm{c}]$ // Get per-center learning rate
         $\mathbf{c} \leftarrow (1 - \eta)\mathbf{c} + \eta\mathbf{x}$ // Take gradient step
     end for
 end for
```

B. FILTERING

To make the system more efficient we filter the web pages to eliminate the unwanted or redundant web pages. We use four types of filtering processes (a) Remove common English words (b) Remove Location names (c) Remove single named entities (d) Remove only last names.

C. INDEXING

The web pages are indexed based on the ranks given by the search engine.

5. ACKNOWLEDGMENT

We sincerely thank the faculty of Computer Engineering Dept. at St. Francis Institute of Technology for giving us the chance as well as the support to write this paper. Without their active involvement and the right guidance this would not have been possible.

6. REFERENCES

[1] Rabia Nuray-Turan, Zhaoqi Chen, Dmitri V. Kalashnikov, Sharad Mehrotra, “Exploiting Web querying for Web People Search in WePS2” : 18th WWW Conference, April 2009.

[2] Javier Artiles, Julio Gonzalo, Satoshi Sekine, “WePS 2 Evaluation Campaign: Overview of the Web People Search Clustering Task”, Appeared in ACM SIGIR Conference, July 2008.

[3] Wahiba Ben Abdessalem Karaa, “Named Entity Recognition Using web Document Corpus”, International Journal of Managing Information Technology (IJMIT) Vol.3, No.1, February 2011.

[4] Wei Lai, Xiaodi Huang, Ronald Wibowo, and Jiro Tanaka, “An On-Line Web Visualization System with Filtering and Clustering Graph Layout”, IEEE Intelligent Informatics Bulletin (2005), Volume: 5.

[5] Sculley,”Web-Scale K-Means Clustering”, published in WWW '10 Proceedings of the 19th international conference on World Wide Web