User Profile Identification Based on Text Mining

Petr Grolmus*
indy@civ.zcu.cz

Jiří Hynek**
jiri.hynek@insite.cz

Karel Ježek*
jezek_ka@kiv.zcu.cz

Abstract: Various scientific disciplines at the University of West Bohemia in Pilsen attract researchers in various fields of specialty. Because of numerous locations of University buildings combined with overlapping focus of faculties, it is often the case that people of the same interest, working at one institution, do not know each other, which impedes sharing of their experience. It is our objective to merge experts based on similarity of their user profiles. Another objective is to utilize user profiles to find additional documents that may be of interest of a user. Packet filtering approach is used to collect information on users. User profiles are generated with the aid of Suffix Tree Clustering algorithm on the basis of characteristic phrases.

Key Words: text mining, user profile, web, www, recommender system, expert search, clustering, suffix tree, phrase search, characteristic phrase, similarity, packet filter.

1 Introduction - Objectives

User profile identification project is currently being implemented at the University of West Bohemia. It is the objective of the authors to link people of similar interests working at different university departments, and thus facilitate sharing of information and experience. Main project goals include automatic generation of user profiles and subsequent expert search. We have been designing the system with minimum (preferably zero) user-feedback requirements in mind.

2 Behavioral Patterns of Web Users

In order to create user's personal interest profile, it is necessary to identify his/her favorite pages and articles. Such information can be acquired in various ways.

2.1 Use of Web Server Log Files

Information on users can be gathered from web server log files. This approach entails some disadvantages:

1. Log files are accessible to administrators only – which is the case of departmental servers. However, proportion of connections to "home" servers in the overall users' activity is negligible.

---

* University of West Bohemia, Univerzitní 8, 306 14 Plzeň (www.zcu.cz)

** inSITE, s.r.o., Rubešova 29, 326 00 Plzeň (www.insite.cz)

2. User is identified by IP address only, which will make identification of a particular user difficult, as each PC can be shared by several users (the case of University labs).

Because of the above difficulties, we have ruled out this approach.

2.2 Proxy Cache Logs

This approach to information collection is often mentioned by researchers (e.g. [1] and [2]). Unfortunately, it entails several drawbacks:

1. It does not solve the problem of user's workstation identification by means of IP address.
2. User is **forced** to use *proxy cache* service.

We have ruled out this option as well.

2.3 Use of a Packet Filter

Capturing of information on documents found to be interesting for a user by means of a packet filter is deemed to be the most suitable approach, although it entails one drawback: users are made to launch the packet filter as an external application. Nonetheless, this drawback is balanced by the following benefits:

1. Each user can be clearly identified, e.g. by a number entered while registering to the system.
2. The packet filter can be switched off any time, e.g. to prevent undesirable alteration of user's profile by some documents, or to maintain user's privacy.

3 System's Design

Figure 1 below depicts system's design.

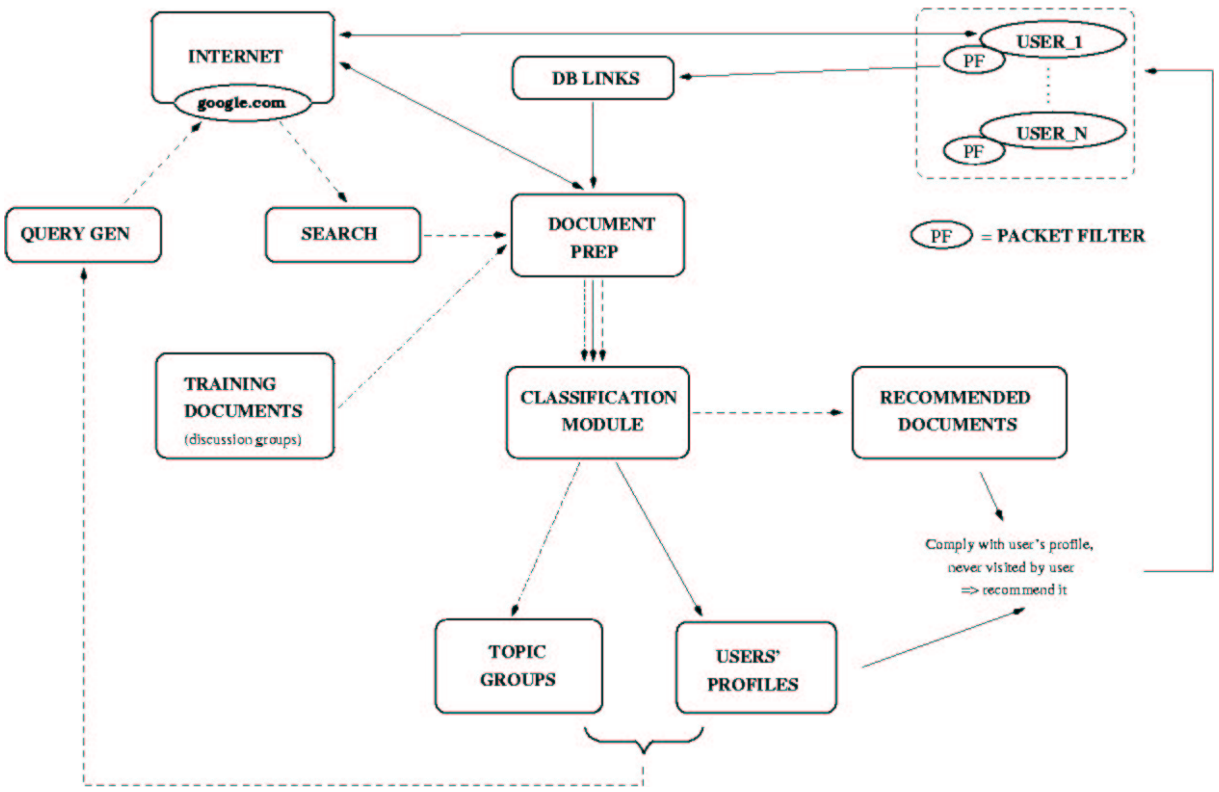

Figure 1: System's design.

Each user is using a packet filter (**PF**) registering all user's requests dispatched via HTTP to Internet servers. User's requests are stored in database (**DB LINKS**) of the central server, waiting there for processing. The first step of document processing (**DOCUMENT PREP**) entails pre-processing of original documents collected from the web into a form acceptable by the system. **CLASSIFICATION MODULE** performs categorization of documents into topics (**TOPIC GROUPS**) based on document contents. Based on contents of documents considered interesting by a user, it is possible to create a user profile and decide which topics may be of user's interest. Utilizing user profile and topics associated with it, a query (**QUERY GEN**) is generated and dispatched to the search server (`www.google.com`) in order to look up documents corresponding to a particular profile. Pages returned by the search server are processed by **SEARCH** module, which identifies recommended web pages in the document. ULR links previously visited by the user are filtered out first. All new URL links are recommended to the user (**RECOMMENDED DOCUMENTS**), as these correspond to user's interests. **TRAINING DOCUMENTS** are designated for a priori generation of topics – these include contents of discussion groups on various topics (computer science, history, astronomy, etc.).

4 Processing of Incoming Documents

4.1 Document Conversion into Plain Text

DOCUMENT PREP module performs conversion of web documents into plain text, eliminating all auxiliary and formatting elements. The resulting text is then passed to **CLASSIFICATION MODULE**. So far we have integrated suitable filters for plain text, HTML, MS Word, postscript and PDF documents. Document types are identified by `file` utility used in *Unix/Linux* systems (note: `file` utility as such is an example of a decision table-based classifier). We are using these conversion utilities:

- HTML – `html2text` (see `http://www.chiark.greenend.uk/ucgi/~richardk/cvswev/html2text`),
- MS Word – `antiword` (see `http://www.winfield.demon.nl/index.html`),
- PDF – files are converted by means of Acrobat Reader (see `http://www.adobe.com`),
- postscript – `ps2ascii` (see `http://www.tardis.ed.ac.uk/~ajed/psutils/`).

4.2 Document Language Identification

Before we proceed to text stemming, it is necessary to identify document language, as stemming process is highly language dependent. We are currently able to identify English and Czech with high confidence, based on occurrence of stop words (despite some stop-word homographs existing in these languages). Please note it is much more difficult to differentiate between Czech and Slovak documents, for example.

4.3 Stemming

A combination of two stemming approaches (dictionary and algorithmic) has been implemented. For dictionary method, we are using publicly available *i-spell* to generate existing word forms (our database for Czech thus includes over 3,3 million word forms). Algorithmic approach that has been implemented (a simplified version of Porter's algorithm – see `http://www.tartarus.org/~martin/PorterStemmer/def.txt`) is based on cutting off the longest possible word endings, maintaining words of at least three characters. Algorithmic stemming is strongly language dependent.

First, we apply the dictionary method. In case a word is not found in the applicable dictionary database, algorithmic stemming is applied.

5 Database Design

5.1 E-R-A Scheme

Database is used to store all information on documents visited by a user, user profile, groups and recommended files. E-R-A scheme is depicted in Figure 2. **USER** table contains information on users collected from a registration form. **VISITED_URL** contains information on URL addresses visited by users. Data in this table are updated continuously by **PACKET FILTER** application. Auxiliary information on each URL address (associated with a user) is also stored, such as the date of the last visit and the total number of visits. **RECOMMENDED_URL** table contains similar information on URLs found to be suitable for a user based on his/her profile generated by the system.

Specific addresses, such as Internet portals and search tools are filtered out automatically, as these could corrupt user's profile. A list of these Internet sources is maintained in **FORBIDDEN_ULR** table. We are using almost 2,000 URLs for this purpose (see `http://www.twics.com/~takakuwa/ search/search.html`).

PHRASE table contains characteristic phrases identified in documents of a particular user. As any document can be visited by several users, document phrases are linked to user records via **PHRASE_USER_URL** table. Each user can be characterized by several clusters **USER_CLUSTER**. Since each cluster consists of a set of phrases, **CLUSTER_PHRASE** table must be used.

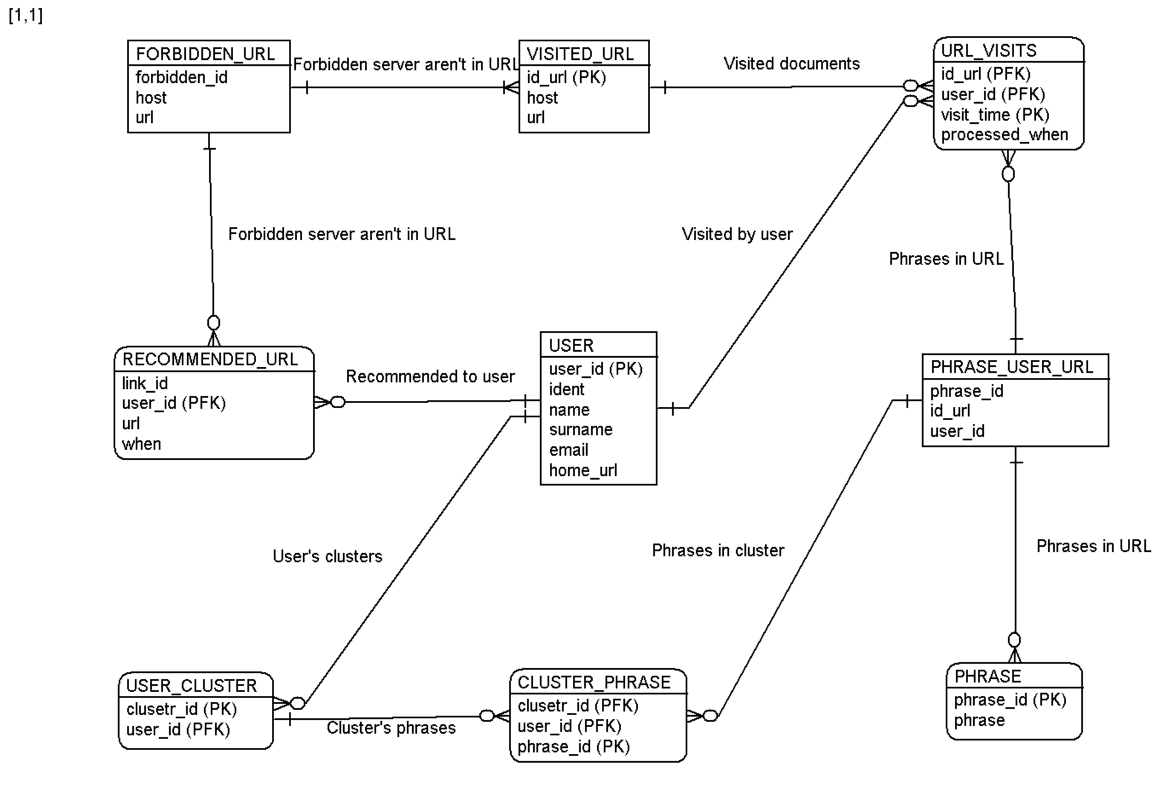

Figure 2: System's E-R-A scheme.

5.2 Aspect of Time

Adaptive user profiles will be used, as user's interests may change in the course of time. We will be using metadata on user's visits (`visit_time` attribute in **`URL_VISITS`** table, see Fig. 2) in order to discard pages not visited for a long time from collection of documents characterizing a particular user. Re-generation of user profiles in six-month intervals seems to be a reasonable choice.

6 Document Encoding

We had to create a mechanism for exchanging documents among individual modules. Therefore, we have created a proprietary format based on XML, describing data on the basis of pre-defined document structure. Here is the proposed DTD for our documents:

```
<!DOCTYPE document_collection [
<!ELEMENT document_collection - - (description?, document+) >
<!ELEMENT description - - (#PCDATA) >
<!ELEMENT document - - (document_id, url?, language?, title?, topiclist?,
keywords?, abstract? content) >
<!ELEMENT document_id - - (#PCDATA) >
<!ELEMENT url - - (#PCDATA) >
<!ELEMENT language - - (#PCDATA) >
<!ELEMENT title - - (#PCDATA) >
<!ELEMENT topiclist - - (topic+) >
<!ELEMENT topic - - (#PCDATA) >
<!ELEMENT keywords - - (#PCDATA) >
<!ELEMENT abstract - - (#PCDATA) >
<!ELEMENT content - - (#PCDATA) >
]>
```

7 Packet Filter

`WinCap` dynamic library is utilized for packet trapping (*the Free Packet capture Architecture for Windows* – see `winpcap.polit.it`). It is capable of packet filtering on operating system's kernel level. Upon its launch, the library can trap all packets of a local workstation, and save these into a buffer for later use. For our purposes we are trapping only those packets that are distributed via port 80 (standard HTTP port) and contain requests to web servers (i.e. `GET` command). From these packets we filter out requests to documents that cannot be processed (files containing images, sound, video, compressed binary data, etc.). We can identify server address (using `host` item) and file path easily.

8 Document Clustering

For document clustering we have implemented *Suffix Tree Clustering* algorithm (see [3], for example) based on frequent phrase search in a document collection. We regard all documents requested by a single user as a special document collection, subject to frequent phrase search. User's profile is then represented by results of this search.

8.1 Suffix Tree Clustering

Suffix Tree Clustering (STC) algorithm is similar to creating an inverted list of phrases occurring in a document collection. Complexity is in the order of $O(n)$ (see [3]), where n is the number of documents in the collection. We will apply the mechanism to all documents requested by a user to be displayed in his/her web browser. Documents will be subject to stemming and stop-word filtering. The maximum length of phrases being searched is $k_{max} = 3$. In fact, this argument defines the length of sliding window used to process text documents.

8.2 Clustering Algorithm Implementation

STC algorithm creates a logical structure corresponding to N-nary tree (see figure 3).

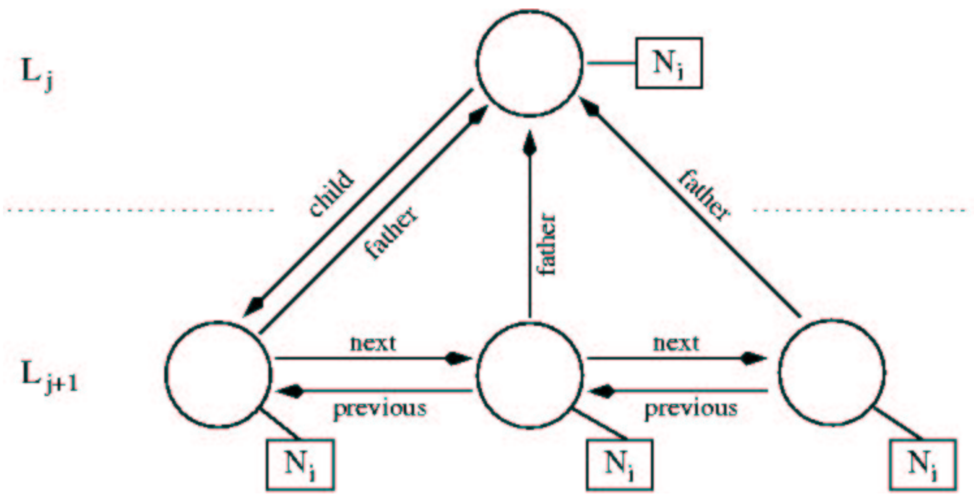

Figure 3: Implementation of STC algorithm.

Each tree node (except root of the tree located at L_0) contains a link to one's "father" (i.e. previous word in a particular phrase). Tree nodes located at the same level (L_j, $j=1,\ldots k_{max}$) with the same father are stored in a linked list. "Child" link goes from a higher-level node to the beginning of a linked list of its successors. Each node i is associated with N_i representing frequency of a phrase in document collection. In order to determine discriminatory power of phrases in document collection, the weight factor is computed for each phrase:

$$w(f_i) = L(f_i) \times N_i,$$

where f_i is a phrase corresponding to node i of STC tree, $L(f_i)$ is its depth, N_i is frequency of its occurrence in document collection.

8.3 Forming of Clusters

We pick *top P* nodes (i.e. the first P nodes with the highest weight, say $P=20$) and consider these characteristic nodes (phrases) of the corresponding document collection (associated with a particular user). We are interested in *similarity* of phrases in order to find *clusters* of phrases (using a mechanism similar to frequent itemsets mining).
Similarity of two phrases can be quantified by means of documents in which these phrases occur. Let's define the following binary similarity measure for phrases f_m and f_n:

$$\frac{|D_m \cap D_n|}{|D_m|} \geq \phi \ AND \ \frac{|D_m \cap D_n|}{|D_n|} \geq \phi \ ,$$

where D_m and D_n represent documents containing f_m and f_n phrases, respectively. ϕ represents threshold with a significant impact on cluster generation.

The longer phrases are used for user profile identification (k_{max}), the lower threshold ϕ for similarity metrics must be selected (and vice versa).

Phrases linked by similarity relationships form a *phrase association graph (**PAG**)*. *Profile clusters* (***PC***) are represented by *maximum complete subgraphs* (i.e. *cliques*) of the phrase association graph. If $\omega(PAG)$ denotes the number of vertices of the largest clique, it also denotes the number of phrases of the largest cluster. The number of cliques represents the number of clusters (i.e. the number of different areas of user's interest). We can define another useful threshold for the minimum acceptable size of a clique (e.g. three vertices/phrases or more). The upper limit for $\omega(PAG)$ is P (very unlikely case). Value of $\omega(G)$ is also known as "*cliqueness*" of a graph G.

The higher the value of ϕ threshold is, the fewer edges of ***PAG*** are formed, decreasing the average size of profile clusters, while increasing the number of these clusters. We have not specified any upper limit for the number of clusters yet, although it is reasonable to believe, that one user will not be characterized by more than three or four identifiable areas of interest.

Alternatively, we can measure phrase-to-phrase similarity by means of *support* and *confidence*:

$$\text{support:} \quad \frac{|D_m \cap D_n|}{|D|} \geq \phi_{\text{sup}} \ ,$$

$$\text{confidence:} \quad \frac{|D_m \cap D_n|}{|D_m|} \geq \phi_{conf} \ AND \ \frac{|D_m \cap D_n|}{|D_n|} \geq \phi_{conf} \ ,$$

where $|D|$ denotes the total number of documents collected in order to identify profile of a particular user. The edge of *PAG* between phrases f_m and f_n is then described by $sup_{m,n}$ and $conf_{m,n}$ values. If m, n and $l \in$ vertices (*PAG*) and mn, nl and $ml \in$ edges (PAG), then, under assumption of conditional independence of phrases, we can infer that $0 \leq conf_{m,n,l} = conf_{m,n} \times conf_{n,l} \leq 1$.

9 System-User Interaction

Users can register via web browser by entering their name and e-mail. Users download a packet filter (currently available for *Windows* only) and configuration file required for identification. The issue of immoral Big Brother monitoring is handled by registering via the web interface and explicit launch of the packet filter.

In principle, user's feedback in the course of actual work is automatic – user visits a recommended web page, which is registered by the packet filter and passed to evaluation modules. Based on automatic feedback we can modify weights of documents dynamically.

10 Practical Tests

We decided to use RCV1 collection (*Reuters Corpus Volume 1,* see `http://about.reuters.com`) for STC algorithm testing. A specific class (associated with a particular user) is subject to testing by randomly selecting approx. 2/3 of documents of this class (*training documents*), searching for frequently occurring phrases, forming *phrase association graphs (PAGs)*, and defining applicable *profile clusters*. Remaining documents are mixed with a certain number of documents from other categories, and then we try to find profile-cluster phrases in these testing documents. For searching purposes we create variable-sized vector of characteristic phrases (*phrase-centroid*) representing each profile cluster (*PC*). Now we can compute cosine-metric similarity between a testing document D_j and characteristic phrases of profile cluster C_i ($i \in [1 \ldots number\ of\ cliques\ of\ PAG]$):

$$Sim(C_i, D_j) = \frac{\sum_{h=1}^{H_i} \left(w(f_h) \times d_{jh} \right)}{\sqrt{\left(\sum_{h=1}^{H_i} \left(w(f_h) \right)^2 \times \sum_{h=1}^{H_i} \left(d_{jh} \right)^2 \right)}} ,$$

where $Sim(C_i, D_j)$ denotes similarity of cluster C_i and testing document D_j, H_i is the number of characteristic phrases in profile cluster C_i (vector size), $w(f_h)$ denotes the weight of the corresponding characteristic phrase f_h, and d_{jh} represents the number of occurrences of phrase f_h in document D_j. If $Sim(C_i, D_j) \geq \tau$, then $D_j \in C_i$, where τ is a specific threshold.

11 Project Summary

We can summarize the methodology detailed in above paragraphs in form of several steps:

1. Collect information on users (a collection of visited web pages per user);
2. Find frequent phrases (apply STC algorithm);
3. Select characteristic phrases (likely a fixed number per each user);
4. Associate "similar" phrases to form *profile cluster(s)* for each user – formation of *phrase centroids;*
5. Apply results of the above:
   a) Look for similar profile clusters (identify users of similar interests);
   b) Recommend new (non-visited) documents that may be of interests of a user.

12 Conclusions

Our research in the field of user-profile identification at the University is currently in the first phase. We assume that we will be able to make our design more efficient, thus achieving higher precision while measuring user-profile similarity. We are planning to test the clustering method on several document collections in various languages. Our research will also benefit from a practical comparison of various clustering approaches.

This work has been partly supported by grants No. MSM 235200005 and ME494.

References

1. P. K. Chan: *Constructing Web User Profiles: A Non-invasive Learning Approach*. Proceedings of Web Usage Analysis and User Profiling – International WEBKDD'99 Workshop San Diego, U.S.A.
2. R. Koval, P. Návrat: *Intelligent Support for Information Retrieval in the WWW Environment*; Proceedings of Advances in Databases and Information Systems – East European Conference, ADBIS 2002, Bratislava, Slovakia
3. O. Zamir, O. Etzioni: *Web Document Clustering: A Feasible Demonstration*; retrieved from CiteSeer (in February 2003) `http://citeseer.org/`