WEB USER PROFILE USING XUL AND INFORMATION RETRIEVAL TECHNIQUES

Dan MUNTEANU

"Dunărea de Jos" University of Galati
Faculty of Computer Science
Department of Computers and Applied Informatics
111 Domnească Street, 800201-Galati, Romania
Phone/Fax: (+40) 236 460182; (+40) 236 461353
E-mail: dan.munteanu@ugal.ro

Abstract: This paper presents the importance of user profile in information retrieval, information filtering and recommender systems using explicit and implicit feedback. A Firefox extension (based on XUL) used for gathering data needed to infer a web user profile and an example file with collected data are presented. Also an algorithm for creating and updating the user profile and keeping track of a fixed number k of subjects of interest is presented.

Keywords: web user profile, Firefox extension, term weighted vector space model, document representation

1. INTRODUCTION. USER MODELLING

In the future, on-line software systems will play the role of service providers that we know today (e.g. banks, public services, etc.). The potential group of users for these systems will be very diverse, ranging from people that want a simple question answered to people who need specialized professional advice. We aim at intelligent human-computer interaction, which provides responses appropriate for specific users.

User modelling has been involved with the problem of adjusting the system based on the user's characteristics, needs, preferences or habits. So the system holds a "picture" of its user, a profile, which is necessary for the adaptation process. "Adaptation" can be interpreted in several ways in this context. It can be seen as adapted interface, personalized filtering of information, adapted interaction, personalized recommendations, etc. Indeed a word that is being used very often lately is "personalization", which probably explains the purpose of all these systems: to treat the user as an individual, with a unique personality. Several issues are involved in developing a user modelling system, including representation of the user model, its acquisition, its maintenance, and its application to achieve personalization.

The hypothesis from which will start is that exists a person, with a problem, goal, which finds himself in a problematic situation and in order to solve the problematic situation he uses an external information resource. Through a mediator, the person interacts with the information resource, in order to resolve the problematic situation.

The components of the Information Retrieval System are:

- user, with goals, problem;
- information resource, with information objects selected, organized and represented;
- mediator, who helps to support the interaction between user and information resource.

This paper was recommended for publication by Cornelia Tudorie

The goals (Belkin, 1997) of the Information Retrieval System are to support the user in:

- managing the problem, achieving the goal;
- appropriate resolution of the problematic situation;
- effective interaction with information objects/resources;
- finding relevant information.

There are two main types of IR Systems:

- Retrospective, or ad hoc IR (models "one-time" information seeking episode, short-term "information need");
- Information Filtering or routing (models recurrent, regular information seeking, long-term "information need").

User modelling can be used in information filtering and retrieval systems to improve the representation of a user's information needs. User models can be constructed by hand, or learned automatically based on feedback provided by the user about the relevance of documents that they have examined. By observing user behaviour (Kim and Oard, 1998), it is possible to infer implicit feedback without requiring explicit relevance judgments. Previous studies based on Internet discussion groups (USENET news) have shown reading time to be a useful source of implicit feedback for predicting a user's preferences.

Studies have shown (Konstan, et al., 1997) that such explicit feedback from the user is clearly useful but obtaining explicit feedback would likely be problematic in many information access applications because this process is time consuming and so implicit feedback seems a better solution.

2. IMPLICIT FEEDBACK FOR USER MODELLING

Implicit feedback has only an indirect relationship to the user's view of the usefulness of any document. But because it can be collected ubiquitously and in great quantities, the potential impact of implicit feedback might be even greater than that of explicit feedback. For example, InfoScope, a system for filtering Internet discussion groups (USENET), utilized both implicit and explicit feedback for modelling users (Stevens, 1993). The used sources of implicit feedback were: a message was read or ignored, it was saved or deleted. In his study, Stevens observed that implicit feedback was effective for tracking long-term interests because it operates constantly without being intrusive.

Morita and Shinoda (Morita and Shinoda, 1994) introduced another source, proposing an information filtering technique based on observations of reading time. From their study we can say that if a document that the user read for more than 20 seconds is used as relevant, this fact actually produced better recall and precision in an information filtering simulation than using the document explicitly rated by the user as relevant.

The following table presents a framework developed by Oard and Kim (Oard and Kim, 1998).

Table 1. – Implicit feedback framework

Category	Observable user behaviours
Examination	Examination time Editing document
Retention	Saving a document Saving a reference Print document Delete
Quote	Copy-Paste
Annotate	Explicit evaluation

In another research, Kim and Oard (Kim, *et al*, 2000) show that:

- on average, users spend more time reading relevant full-text journal articles than non-relevant articles;
- on average, users spend more time reading abstracts of relevant journal articles than abstracts of nonrelevant articles;
- the combination of reading time and printing behaviour will be more useful for predicting explicit ratings than using reading time alone.

3. FIREFOX EXTENSION

Extensions add new functionality to Mozilla applications such as Firefox and Thunderbird. They can add anything from a toolbar button to a completely new feature. They allow the application to be customized to fit the personal needs of each user if they need additional features, while keeping the applications small to download.

Extension technologies

- XPI — Cross-Platform Installer module
- JavaScript – The primary language of Mozilla browsers
- XUL (XML User Interface Language) – Used to define the UI (User Interface) and interaction with user.
- DOM (Document Object Model) – Used to change XUL in real-time or to edit HTML that is currently loaded
- CSS (Cascading Style Sheets)

- XPCOM/XPConnect

Firefox extensions are generally used to add functions to the browser. Examples of functions which an extension might add include RSS readers, bookmark organizers, toolbars, website-specific client programs, FTP, e-mail, mouse gestures, proxy server switching, or debugging tools. Many extensions can change the content of a webpage, not on the webpage itself, but as it is displayed in an individual user's browser.

The next section presents a Firefox extension (UserProfile) used for gathering data for web user profile.

```
+- userprofile/
  +- install.rdf
  +- chrome.manifest
  +- chrome/
    +- content/
      +- userprofile.xul
      +- userprofile.js
```

The installer manifest (install.rdf) provides details on the extension to Firefox, the chrome manifest (chrome.manifest) tells Firefox what packages and overlays the extension provides. The user interface portion of a Firefox extension (userprofile.xul) is created using XUL, a markup language used in creating user interfaces, which can be thought of as a flavour of XML, simply because XUL is nothing more than XML that makes use of predefined elements (also called widgets). The beauty of XUL comes through the use of what it calls dynamic overlays. A dynamic overlay allows a developer to modify the behaviour of a window's user interface, without having to change the original interface's code. Firefox extensions are usually driven by JavaScript programming language, attaching logic to widgets that execute some function as a result of user input (userprofile.js).

This Firefox extension for creating a user profile keeps track of all the web pages (URLs) that are displayed in each tab, the time spent on each page (milliseconds), the downloaded files, the saved bookmarks and the printed pages. All this information will be used to infer a web user profile model. Some will be positive examples (saved pages, saved bookmarks, web pages displayed for a long time) - others will be negative examples (web pages closed very soon, deleted bookmarks).

Next will be presented some of the information gathered with this Firefox extension:

```
<session timestart="1212851680843">
<webpageprint url="http://www.yahoo.com/"/>
<webpagedownload file="C:\ Yahoo.htm"
url="http://www.yahoo.com/"/>
<webpagevisited
url="http://developer.mozilla.org/en/docs/XUL"
timespent="6954"/>
<webpagevisited url="http://www.yahoo.com/"
timespent="21125"/>
<webpagevisited url="http://java.sun.com"
timespent="17812"/>
<webpagevisited url="http://www.eclipse.org"
timespent="3141"/>
<webpagevisited url="http://news.bbc.co.uk/"
timespent="2390"/>
<webpagevisited url="http://www.cnn.com/"
timespent="4907"/>
<webpagevisited url="http://news.cnet.com/"
timespent="4421"/>
<webpagevisited url="http://java.sun.com/"
timespent="35204"/>
<webpagevisited
url="http://java.sun.com/javase/downloads/?intcmp
=1281" timespent="6343"/>
<bookmarkadd url="http://news.cnet.com/"
name="Technology news - CNET News.com" />
<bookmarkdelete
url="http://www.dinkumware.com/" />
</session>
```

4. WEB USER PROFILE ALGORITHM

The algorithm for creating web user profile is presented below. It keeps track of a fixed number *k* of clusters (categories, subjects of interest). At the beginning, for each of the first *k* documents offered as feedback a new cluster is created. After *k* clusters have been created, each new document is treated as the *k+1* cluster and the algorithm goes on with the merging of the two most similar clusters for keeping the clusters number equal with *k*. That is why a new document can be placed in a cluster which already exists or it can form its own cluster, forcing more similar existing clusters to merge.

The pseudocod for this algorithm is presented below.

```
UpdateProfile(doc)
    cluster [k+1] = newcluster (doc);
    (i, j) = the most similar clusters, with i < j;
    new_cluster = Combine_Cluster (cluster i,
cluster j);
    remove(cluster i);
    remove(cluster j);
    insert(new_cluster).
```

The base idea of the algorithm is to represents each document as a vector and documents with similar content to have similar vectors. Each space dimension represents a term and his associated weight. The values of the weights of the terms from each vector for a document are computed using TF-

IDF (Term Frequency - Inverse Document Frequency) algorithm. In TF-IDF algorithm the weight of a term is the product of the term frequency (TF(*d*,*t*) - number of times the term *t* appear in document *d*) and inverse document frequency(IDF(t)). The inverse document frequency is obtained from the document frequency (DF(*t*) - number of documents in which the term *t* appear at least once). Inverse document frequency is:

$$IDF(t) = \log\frac{N}{DF(t)},$$

where *N* is the total number of documents.
The weight of the term t (W(d,t)) is

$$W(d,t) = TF(d,t) \times IDF(t)$$

Value $v(i)$ of each element in the vector is the product: $v(i) = TF(d,t_i) \times IDF(t_i)$, $i = 1,...,n$ where *n* is the number of terms in the profile vector.

The algorithm for constructing the user profile is presented below. Some assumptions are made. The number of categories (domains of interest) is fixed and is set to *m*. The profile is a set of *m* TF-IDF vectors *V*. The numbers of elements contained in each vector is *n*.

For each positive example (HTML document) the following steps are applied:

- Processing document (HTML page):
  - extract text from HTML;
  - remove STOP words;
  - apply stemming method (e.g. Porter's algorithm)and keep the root of the term.
- Build TF-IDF vector for the document - V_k
- If $|V| < m$ (|V| is the number of vectors in the user profile V) then $V \Leftarrow V \cup V_k$,
- else, compute similarity for each 2 TF-IDF vectors from the profile V including the vector for the new document V_k.

Profile V is $\{V_1, V_2, ..., V_m, V_k\}$

The simmilarity measure used is the cosine measure which representing the cosine of the angle formed by the two vectors in the *m*-dimensional space.

$$sim(V_i, V_j) = \cos(\hat{V_i, V_j}) = \frac{V_j \cdot V_k}{|V_j| \times |V_k|}$$

$$i, j \in \{1,2,\ldots,m,k\}$$

- Combine the two vectors with highest similarity (with indices q and p):

$$V_k = V_q + V_p,$$

where $(q, p) = \arg\max_{(i,j)}(sim(V_i, V_j))$

$$i, j \in \{1,2,\ldots,m,k\}$$

The number of elements in V_k is now 2n.

- sort weights in the new vector V_k in decreasing order and keep the first *n* elements.

5. CONCLUSIONS

This paper began with explaining what user modelling is and showing the importance of using user profile in information retrieval, information filtering and recommender systems. Then relevance feedback is presented with its two forms explicit and implicit feedback. The accent was set on the implicit feedback presenting a framework for implicit feedback. The next part presented a Firefox extension (based on XUL) used for gathering data needed to infer a web user profile (keeps information about all visited web pages, the time spent on each page, the downloaded files, the saved bookmarks and the printed pages) and an example file with collected data. Finally the algorithm for creating and updating the user profile and keeping track of a fixed number k of subjects of interest is presented (Information Retrieval techniques are used for weighting the terms on each user profile and for similarity measure).

REFERENCES

Belkin , N. J. (1997). *User Modeling in Information Retrieval. Sixth International Conference on User Modelling (UM97)*, Chia Laguna, Sardinia, Italy.

Kim J., D. W. Oard and K. Romanik (2000). *User Modeling for Information Access Based on Implicit Feedback. Intelligent User Interfaces*.

Konstan, J. A., B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon and J. Riedl (1997). *GroupLens: Applying collaborative filtering to Usenet News*. *Communication of the ACM* 40.

Morita, M. and Y. Shinoda (1994). *Information filtering based on user behaviour analysis and best match text retrieval*. Proceedings of the *17th ACM Annual International Conference on Research and Development in Information Retrieval (SIGIR'94)*. Dublin, Ireland.

Oard, D.W. and J. Kim (1998). *Implicit Feedback for Recommender System*. *AAAI Workshop on Recommender Systems*. Madison, 1998.

Stevens, C. (1993). *Knowledge-Based Assistance for Handling Large, Poorly Structured Information Spaces*. Ph.D. Dissertation, University of Colorado at Boulder Technical.