Khribi, M. K., Jemni, M., & Nasraoui, O. (2009). Automatic Recommendations for E-Learning Personalization Based on Web Usage Mining Techniques and Information Retrieval. *Educational Technology & Society*, *12* (4), 30–42.

Automatic Recommendations for E-Learning Personalization Based on Web Usage Mining Techniques and Information Retrieval

Mohamed Koutheaïr Khribi[1], Mohamed Jemni[1] and Olfa Nasraoui[2]
[1]Technologies of Information and Communication Lab, Higher School of Sciences and Technologies of Tunis, University of Tunis, Tunisia // mk.khribi@uvt.rnu.tn // mohamed.jemni@fst.rn.tn
[2] Knowledge Discovery & Web Mining Lab, University of Louisville, USA // olfa.nasraoui@louisville.edu

ABSTRACT

In this paper, we describe an automatic personalization approach aiming to provide online automatic recommendations for active learners without requiring their explicit feedback. Recommended learning resources are computed based on the current learner's recent navigation history, as well as exploiting similarities and dissimilarities among learners' preferences and educational content. The proposed framework for building automatic recommendations in e-learning platforms is composed of two modules: an off-line module which pre-processes data to build learner and content models, and an online module which uses these models on-the-fly to recognize the students' needs and goals, and predict a recommendation list. Recommended learning objects are obtained by using a range of recommendation strategies based mainly on content based filtering and collaborative filtering approaches, each applied separately or in combination.

Keywords

E-learning, Automatic Personalization, Recommender Systems, Content based filtering, Collaborative Filtering

Introduction

Up to the very recent years, most e-learning systems have not been personalized. Several works have addressed the need for personalization in the e-learning domain. However, even today, personalization systems are still mostly confined to research labs, and most of the current e-learning platforms are still delivering the same educational resources in the same way to learners with different profiles. In general, to enable personalization, existing systems used one or more types of knowledge (learners' knowledge, learning material knowledge, learning process knowledge, etc). Generally, personalization in e-learning systems concerns: *adaptive interaction*, *adaptive course delivery*, *content discovery and assembly*, and *adaptive collaboration support*. The category of *adaptive course delivery* represents the most common and widely used collection of adaptation techniques applied in e-learning systems today. Typical examples include dynamic course re-structuring and adaptive selection of learning objects, as well as adaptive navigation support, which have all benefited from the rise of using recommendation strategies to generate new and relevant links and items. In fact, one of the new forms of personalization in e-learning environment is to give recommendations to learners in order to support and help them through the e-learning process.

A number of personalized systems have relied on explicit information given by a learner (demographic, questionnaire, etc) and have applied known methods and techniques of adapting the presentation and navigation (Chorfi et al., 2004). As explained in (Brusilovsky, 1996), two different classes of adaptation can be considered: *adaptive presentation* and *adaptive navigation support*. Later, in (Brusilovsky, 2001), the taxonomy of adaptive hypermedia technologies was updated to add some extensions in relation with new technologies. Then, the distinction between two modes of adaptive navigation support became a necessity, especially with the growth of recommender systems. Automatic recommendation implies that the user profiles are created and eventually maintained dynamically by the system without explicit user information. Examples include amazon.com's personalized recommendations and music recommenders like Mystrand.com in commercial systems (Mobasher 2006), smart recommenders in e-learning (Zaiane, 2002), etc. In general, such systems differ in the input data, in user modeling strategies, and in prediction techniques. Several approaches for automatic personalization have been reported in the literature, such as content-based or item-based filtering, collaborative filtering, rule-based filtering, and techniques relying on Web usage mining, etc (Nasraoui, 2005). Web recommender systems can be categorized depending on these approaches. Content-based filtering (or item-based filtering) systems recommend items to a given user based on the correlation between the content of these items and the preferences of the user (Meteren et al., 2000). This means that the recommended items are considered to be similar to those seen and liked by the same user in the past. Thus, there is no notion of a *community* of users, rather only *one* user profile is considered while

making recommendations. Classical examples of systems applying content based filtering approach include among other Personal webwatcher (Mladenic , 1996), syskill and webert (Pazzani et al., 1997), etc.

Collaborative filtering system recommends items that are liked by other users with similar interests. Thus, the exploration of new items is assured by the fact that *other similar* user profiles are also considered. Examples of such systems include GroupLens (Konstan et al., 1997) and (Sarwar et al., 1998). Hybrid recommender systems combine *several* recommendation strategies to provide better performance than either strategy alone. Most hybrids work by combining several input data sources or several recommendation strategies. There are many hybridization methods reported in the state of the art, content/collaborative hybrids are the most popular hybrid strategies. Generally, web recommender systems tend to use web mining techniques in one or more stage of the recommendation process. In e-learning, the interest in using web mining has recently increased, especially with the rapid spread of web based learning environments in education and the growing need to give personalized services to students. In e-learning systems, web mining techniques are used to learn all available information about learners and build models to apply in personalization. A detailed description about using and applying educational data mining was given in (Romero et al., 2006) and (Romero et al., 2007). Several techniques could be used for personalization and recommendation such as classification, clustering, prediction, association rule, and sequential pattern. In (Tiffany et al., 2003), students were clustered with similar learning characteristics and applied collaborative filtering to provide paper recommendation.

A recommender agent using association rules has been used to recommend e-learning activities in (Zaiane, 2002). Providing to a student the next link or task to do within the adaptable educational hypermedia system AHA! was the aim of the recommender system described in (Romero et al., 2007). In this paper we are going to describe an automatic personalization approach for providing learning object recommendations for online students in e-learning systems. It is to be noted here that by adopting the term "Learning Object" we mean any digital educational resource used within the e-learning environment, it could be a course, a web page, a simulation, i.e. all known formats of digital educational resources regardless their granularity. These learning objects are referenced within the e-learning system by their URL, they are also archived in log files or tracked in databases as URL references. Therefore, automatic recommendation of learning objects means generating a list of URLs referencing educational resources (hosted and/or created inside the e-learning platform) in order to guide and support the e-learners. The proposed approach is taking into account both the Web access history of learners as well as the content of the learning material, Web mining techniques in combination with an open source Web information retrieval system are used to enable an implementation that is not only open and scalable, but also fast to deploy. The recommender system we aim to develop should be considered as an external module or plug-in that can be included easily in e-learning systems (courseware, LMS, etc) to give automatic personalization. This paper is arranged in the following way : first we describe the proposed approach and the corresponding phases of modeling and recommending. In Section 3, we present some implementations of the proposed methodologies. In Section 4, we make some experiments and evaluation. Finally, conclusions and future work are presented.

A framework for building automatic recommendations in e-learning platforms

Our proposed framework is composed of two modules: an off-line module which pre-processes data to build learner and content models, and an on-line module which uses these models on-the-fly to recognize student goals and predict a recommendation list. Recommended learning objects are obtained by using a range of recommendation strategies based mainly on content based filtering and collaborative filtering, each applied separately or in combination. The recommendation procedure is performed using the following tasks:

- **Preliminary offline mining of learners' models** based on Web usage mining techniques. First, we gather web learners' sessions and we apply a clustering approach to directly cluster these sessions. Each cluster contains similar sessions, showing similar interests of different learners. Each cluster can also be viewed as one learner's model;
- **Preliminary offline mining of association rules** (e.g. "Resource A → Resource B") from clustered sessions;
- **Preliminary offline crawling and indexing of learning resources**: this step consists of crawling the entire learning resources available in a course repository and forming an *inverted* index mapping each keyword to a set of pages in which it is contained;
- **Extracting user preferences from the learner's active session** (set of URLs or list of terms extracted from these URLs);

- **Computing relevant links to recommend for the active learner** by applying a number of recommendation strategies.

The proposed approach, with main features depicted in (Figure 1), is essentially based on two components: *the Modeling phase* and the *Recommendation phase* (Khribi et al., 2008).

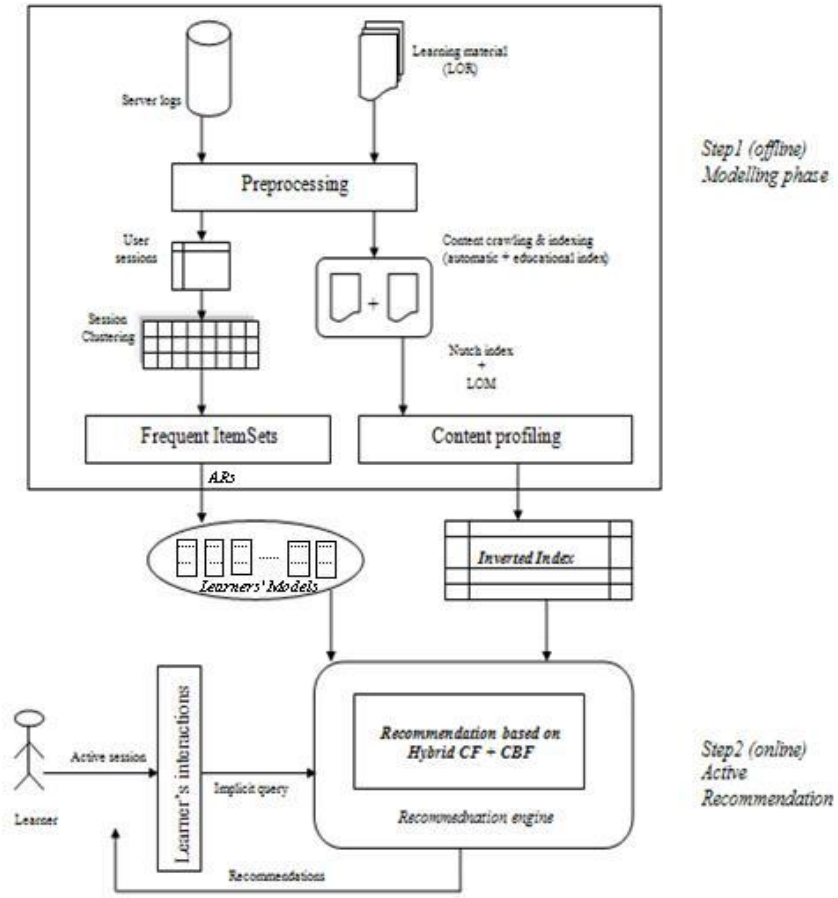

Figure 1: Proposed personalization approach

Modeling phase

Learner Modeling: Despite the availability of abundant educational resources and services, it is still difficult to decide which learning objects better match the student's needs in a given situation, unless we accurately know the profile of that particular learner and his/her online behaviour. A learner's model is composed by a range of relevant information about the student using the e-learning environment. Frequently common types of information used in learners' models can include the learner knowledge, demographic information, preferences, learning styles, etc. These components are strongly connected to the application of the learner model. Generally, according to (Brusilovsky, 1994), two main categories are outlined : the domain specific information and the domain independent information. In our approach, we intend adopting a three components learner's model : learner's profile, learner's knowledge, and learner's educational preferences. Regarding the scope of this paper, we use solely the learner's knowledge to represent the learner's model. Other parts are being processed and are resumed in further papers. The learner's model (reduced here to his/her learner's knowledge component) can be represented by a sequence of weighted visited learning objects i.e. a vector of visited learning objects or curriculum elements in which the student was interested. Most approaches for user modeling heavily depend on user feedback. Users with common interests and levels can be grouped together. Feedback from one user can serve as a guide for information delivery to the other

users within the same group. The learner's model is generally built based on a range of information, gathered through the user's implicit and/or explicit feedback, describing the learner's preferences on learning objects.

Basically, two main learner modeling approaches can be outlined: collaborative learner modeling and automatic learner modeling. The collaborative learner modeling approach requires students to provide explicit information about their preferences and needs. In the automatic learner modeling approach, gathering information is done rather automatically based on the online behavior and activities (i.e. implicit feedback) of students. Automated building of learner models involves the automated detection of all basic information composing the model i.e the learner's knowledge in our case. Indeed, huge amount of data are collected continuously from the student interactions and browsing history, and capitalized automatically on server's side and/or e-learning system database. To analyze these tracked data and make it fruitful to use we apply Web mining techniques. Data must be first pre-processed, using tasks including: data cleaning, user identification, and session identification. Corresponding to each student, we obtain set of sessions (i.e. several sessions gathered over a period of time). Let LO be a set of n unique visited learning objects : $LO = \{ LO_1, LO_2, LO_3, \ldots, LO_n\}$, and let L be a set of m learners registered in a specific course within the e-learning environment, $L= \{L_1, L_2, \ldots, L_m\}$, the learner knowledge model LK_i corresponding to the learner $L_i \in L$ is represented by a set of p sessions S_j^i extracted from tracked data : $LK_i = \{S_1^i, S_2^i, S_3^i, \ldots, S_p^i\}$, where each S_j^i is a subset of k weighted visited LO_i,

$S_j^i = < (LO_1^{S_j^i}, w(LO_1^{S_j^i})), (LO_2^{S_j^i}, w(LO_2^{S_j^i})), (LO_3^{S_j^i}, w(LO_4^{S_j^i})), \ldots, (LO_k^{S_j^i}, w(LO_k^{S_j^i})) >$, where each $LO_k^{S_j^i} = LO_l$ for some $l \in \{1, \ldots, n\}$, and $w(LO_k^{S_j^i})$ is the weight associated with learning object reference $LO_k^{S_j^i}$ in the session S_j^i corresponding to the i^{th} student L_i. The knowledge model of a learner L_i (i.e learner model) can be represented by a matrix $M(p, n)$ where p is the number of completed sessions and n the cardinality of unique visited learning objects :

$$\begin{pmatrix} w(LO_1^{S_1^i}) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ w(LO_1^{S_p^i}) & \cdots & w(LO_n^{S_p^i}) \end{pmatrix}$$

Group Modeling: Once learners' models are delimited properly, we apply a two-level model based collaborative filtering approach in order to organize the obtained models into groups of learners based on similarities and dissimilarities among their preferences. In the first level, we apply clustering techniques based on similarities and dissimilarities among preferred visited learning objects. A variety of clustering techniques can be used for clustering sessions. (Shahabi et al., 1997) described a prototype system that uses viewing time as the primary feature to describe a user session and then clusters the sessions using K-Means clustering. (Mobasher et al., 2000) used a multi-variate K-Mmeans algorithm to obtain transaction clusters and the Association Rule Hypergraph Partitioning (ARHP) technique to obtain usage clusters. (Yao et al., 2002) applied the Leader algorithm for clustering. In (Zhao et al., 2005), hierarchical clustering algorithm is presented and applied in *CLUTO* software for clustering high-dimensional datasets. Regardless of which method is used, the first clustering level will result in a set $C=\{C_1, C_2, \ldots, C_k\}$ of clusters where each cluster C_i^j a subset of student sessions representing a group of similar learners with similar access patterns. It should be noted here that the similarity measure used in clustering is the cosine similarity as shown in (1). In this equation, we have considered binary weight i.e. existence or non existence of a referred learning object in a session.

However, to give to the similarity measure more sense (semantic similarity measure), we can consider learning object content implicitly by taking into account a hierarchical web pages structure as in (2). In (3), $S_u(i,j)$ is a URL to URL similarity function computed based on the amount of overlap between the paths P_i and P_j leading from the root of the website to any two URLs i and j (Nasraoui et al., 2008).

$$s_j^{(i)} = \begin{cases} 1 & \text{if student accessed } j^{th} \text{ LO} \\ 0 & \text{otherwise} \end{cases} \qquad S_{1,kl} = \frac{\sum_{i=1}^{N_U} s_i^{(k)} s_i^{(l)}}{\sqrt{\sum_{i=1}^{N_U} s_i^{(k)} \sum_{i=1}^{N_U} s_i^{(l)}}} \qquad (1)$$

$$S_{2,kl} = \frac{\sum_{i=1}^{N_U}\sum_{j=1}^{N_U} s_i^{(k)} s_j^{(l)} S_u(i,j)}{\sum_{i=1}^{N_U} s_i^{(k)} \sum_{i=1}^{N_U} s_i^{(l)}} \qquad (2) \qquad S_u(i,j) = \min\left(1, \frac{|p_i \cap p_j|}{\max(1, \max(|p_i|, |p_j|) - 1)}\right) \qquad (3)$$

The second level applied on each obtained cluster C_i (containing several sets of referred learning objects) consists of using, first, a frequent itemset mining algorithm that extracts frequently co-occurring referred learning objects in sessions belonging to each cluster. Then, association rules *(AR)* are extracted from these clusters. Association rules capture the relationships among learning objects references based on their co-occurrence across sessions. AR discovery methods such as the Apriori algorithm (Agrawal et al., 1994) can be applied. An association rule r is an expression of the form: A => B, where A and B are itemsets, r must satisfy a minimum confidence threshold to be involved in recommended page set. The confidence α_r of a rule r is given by $\sigma(A \cup B)/ \sigma(A)$ and its support σ_r is the support of $(A \cup B)$.

$$\sigma(I_i) = \frac{\left|\left\{S_l^j \in C_j : I_i \subseteq S_l^j\right\}\right|}{\left|C_j\right|}$$

Thereby, further in recommendation phase, the new active learner is directly classified in one of the discovered groups of students (clusters), then personalized recommended links are provided by matching the learner current navigation with the AR of the corresponding group.

*Content Modeling: G*enerally, content modeling involves applying indexing and text mining *techniques* (which are part of Web content mining). The originalities of our approach are twofold: (1) the use of the open source search engine Nutch (http://lucene.apache.org/nutch) in the content modeling phase, followed by content based filtering as a recommendation strategy. (2) The automated indexing of standard defined *educational* content *thanks* to the search engine's powerful capabilities in the *automated* and *scaled crawling and indexing* phases.
To improve the educational content indexing phase covering two important levels : data (content) and meta-data (LOM), specific index fields used in *LOM (Learning Object Metadata)* are added to the native Nutch index. Thereby, accuracy of the final index, content search and recommendations are improved. Finally, we note that a byproduct of our crawling and indexing with Nutch is an available interface to use for explicit searches over the material if needed. Specific attributes to add into the Nutch index structure are given by the educational metadata providing descriptions and additional information (author, title, learning resource type, technical requirements, rights management, etc) about learning resources. These fields are added using a specific updated plugin. Corresponding information are loaded and inserted automatically to the inverted index thanks to XML files embedded to Standard defined Learning Objects and Nutch capability to crawl and parse XML files.

Recommendation phase

Formulating a learner's implicit query: The learner's implicit query is a set of referred learning objects recently visited by an active learner. Such query should be extracted implicitly from the recent navigation history of the learner and could be represented by a vector of referred learning objects or a vector of relevant terms describing these learning objects. This task is accomplished in two phases (1) delimiting the current active learner session, and (2) extracting URLs (Learning Objects references) of interest from this active session. Since the active user session is to extract from the Web log file (or tracked data in DB), we identify only the records representing the last *W* visited pages called the sliding window. This is done by taking into account the timestamp when a learner connected to the system and the timestamp of a recommendation demand. In order to express with more details learner preferences and interests, we can associate to each URL, composing learner sessions, a given weight. These weights can be binary, representing the existence or non-existence of an URL reference in the session; or they can be computed as a function of a number of features based essentially on the frequency of occurrence of URL within a session and the time a learner spends on a particular page as a manner to determine indirectly the fact that a learner liked or disliked the URL (Shahabi et al., 1997), (Yan et al., 1996). In addition to that, many other features could be added to compute page weight, but, generally, such features are considered as not a good indication of the user interest (Konstan et al., 1997). In the present work, we didn't consider the URL weight. Let *W* be a fixed size for a sliding window, then if the active user session with *W* =3 is {*A*,*B*,*C*}, and the URL "*D*" was viewed last by a user, then the new sliding window becomes {*B*,*C*,*D*}. It should be noted that *W* can lead to lower or higher recommendation coverage. In our case, we consider a fixed window size *W* =3, so only the last three visited pages will affect the recommendation. Once we have obtained the sliding window pages, we translate it into a term vector. In fact, each page is mapped to a set of content terms characterizing it. We developed a java code (based on Nutch's built in functionalities for parsing HTML pages, and a plug-in for stop word elimination) that returns, corresponding to a given page, the top *K*

frequent terms sorted by their frequencies. In our experiment (see section 3), we considered a fixed number of terms K =3. Obtained terms are representing what we call a term vector (Figure 2).

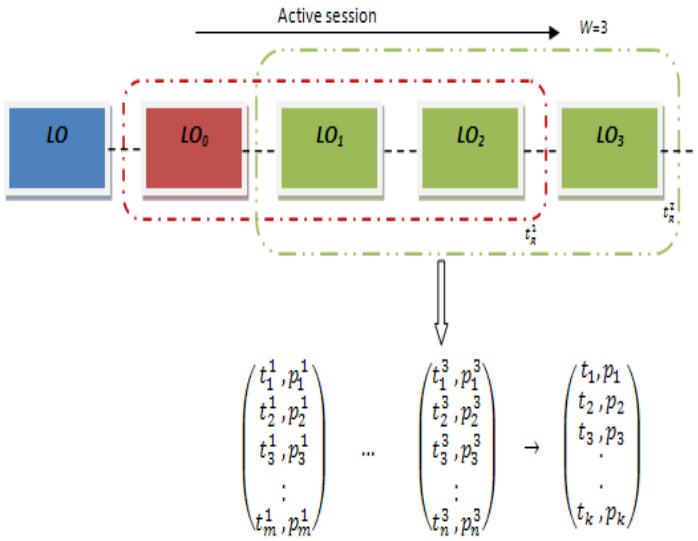

Figure 2: Term Vector Building process

Recommendation process: The learner's implicit query defined previously under both of its shapes constitutes the input of the recommendation phase. The recommendation process task is accomplished using basically : content-based filtering (CBF) and collaborative filtering (CF) approaches (Figure 3). First, we apply the (CBF) approach alone using the search functionalities of the Nutch search engine. We submit the term vector to the search engine in order to compute recommendation links. Results are ranked according to the cosine similarity of their content (vector of *TF-IDF* weighted terms) with the submitted term vector. Second, we apply the collaborative approach (CF) alone by comparing, first, the sliding window pages to clusters (groups of learners obtained in the offline phase by applying two-level model based collaborative filtering approach) in order to classify the active learner in one of the learner's group. Then, we use the *ARs* of the corresponding group to give personalized recommendations. The current session window is matched against the "condition" or left side of each rule. (Example: *"75.43 % of learners who visited http://cours.uvt.rnu.tn/rpl/cours/informatique/bases_donnees/base_donnee/chap6/menu.htm visited http://cours.uvt.rnu.tn/rpl/cours/informatique/bases_donnees/base_donnee/chap6/obje.htm and http://cours.uvt.rnu.tn/rpl/cours/informatique/bases_donnees/base_donnee/chap6/index6.htm"*). All rules should be sorted by their confidence (highest to lowest) when forming the recommendations.

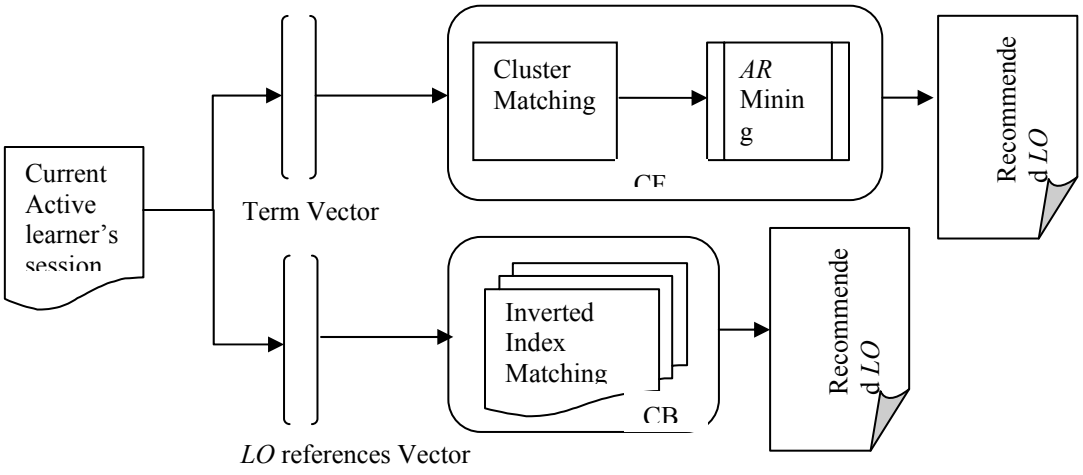

Figure 3: Recommendation process

It is worth noting that several recommendation strategies using these approaches have been investigated in our work. After applying a CF and CBF approaches alone, we included next the possibility to combine *both* of the recommendation approaches (CBF and CF) in order to improve the recommendation quality and generate the most relevant learning objects to learners. Hence, two approaches are to be considered: *Hybrid content via profile based collaborative filtering* with *cascaded/feature augmentation combination*, which performs collaborative recommendation followed by content recommendation (the reverse order could also be considered); and *Hybrid content and profile based collaborative filtering* with *weighted combination*, where the collaborative filtering and content based filtering recommendations are performed *simultaneously*, then the results of both techniques are combined together to produce a single recommendation set (Nasraoui et al., 2006).

In the Hybrid content via profile based collaborative filtering with cascaded/feature augmentation combination approach, we apply first CF approach giving as output a set of recommended links, then we apply CBF approach on these links. In fact, recommended links are mapped to a set of content terms in order to compose a term vector (top *k* frequent terms), a *parser* tool must be used for this task. Finally, these terms are submitted to the search engine Nutch which returns the final recommended links.

In the Hybrid content and profile based collaborative filtering with weighted combination approach, the collaborative filtering and content based filtering are performed *separately*, then the results of both techniques are combined together to produce a single recommendation set. This process uses the following steps:

I. Step 1 is performed in the same way as in CF approach, the result is called *Recommended Set 1* ;
II. Step 2 maps each *LO* references in the sliding window to a set of content terms (top *k* frequent terms). Then these terms are submitted to the search engine which returns recommended links. This result is called *Recommended Set 2;*
III. Final collaborative and content based filtering recommendation combination: both recommended sets obtained previously are combined together to form a coherent list of related recommendation links, which are ranked based on their overlap ratio.

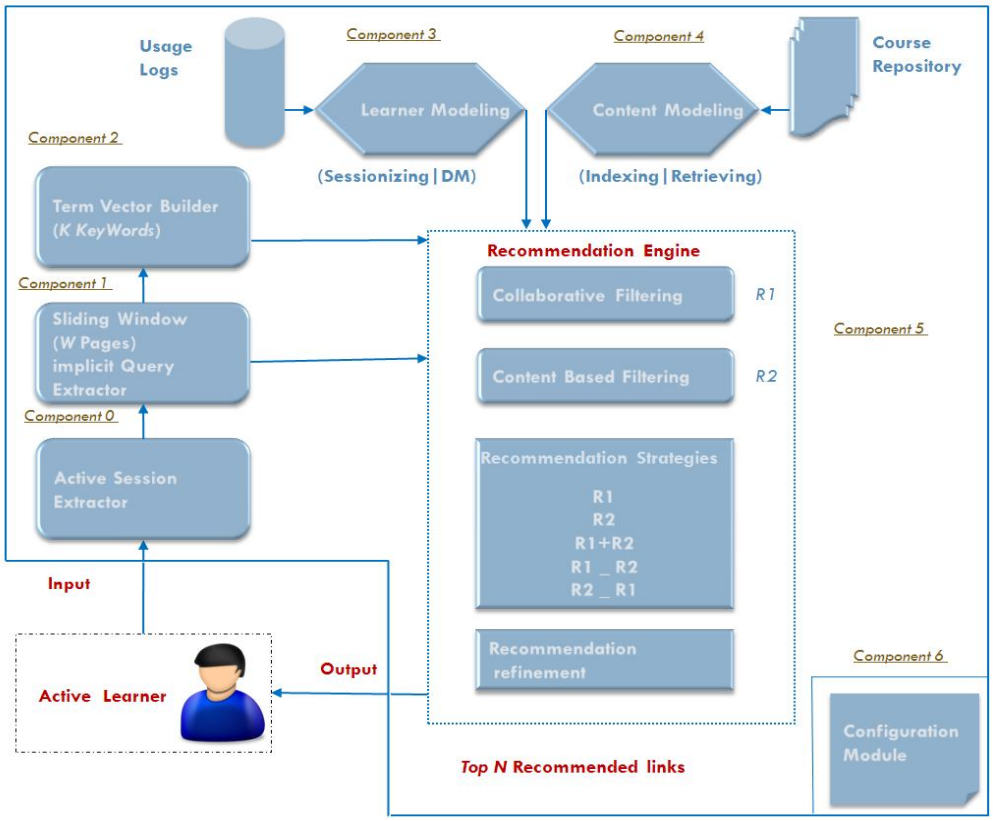

Figure 4: Architecture of the hybrid e-learning recommender system

Experimentation and results

To implement and evaluate the proposed personalization approach, we conceived a system composed by a set of components, where each component is performing a number of services. The main features of the proposed recommender system are shown in Figure 4.

Component 0 extracts the learner active session from the Web log file, starting from the time that the learner connected to the platform until he/she asks for recommendations. *Component 1* and *Component 2* represent the input for the recommender system in two different forms. Component 1 extracts the *W* sliding window pages from the active learner session. The second input form is performed by Component 2 which extracts from Component 1's results (sliding window) the top *K* relevant terms. Each page of the sliding window is transformed into a set of content terms. This task is performed using Nutch's built-in functionalities for parsing HTML pages, and a plug-in for stop word elimination. Components 0, 1 and 2 are performed in an on-line mode. Component 3 concerns the phase of learner modeling which is done in an offline mode. We used the course repository of the Virtual University of Tunis, RPL platform (http://cours.uvt.rnu.tn) as an experimentation environment for our system. Web usage logs are collected from the RPL log files (Apache web access log files http://httpd.apache.org/docs/2.0/logs.html) in the period between February and July 2007. RPL Log files were rotated based on daily directives, thus generating 180 large log files with a total of 3,049,986 requests. The log format was ECLF (Extended Common Log Format), further enriched with added user authentication information to make the session extraction error free (using an embedded session Id mechanism, added via the Apache configuration and our RPL code). Since the collected log data contain many uninteresting elements (graphics, icons, requests generated by crawlers/bots, etc), it must first be pre-processed. Starting with 3,049,986 requests, the cleaning operations resulted in only 594,325 nontrivial requests. Figure 5 shows the variation of request numbers before and after data cleansing per month. After cleaning the original logs, we sessionized the remaining requests. Figure 6 shows the variation of the number of resulting unique usage sessions per month.

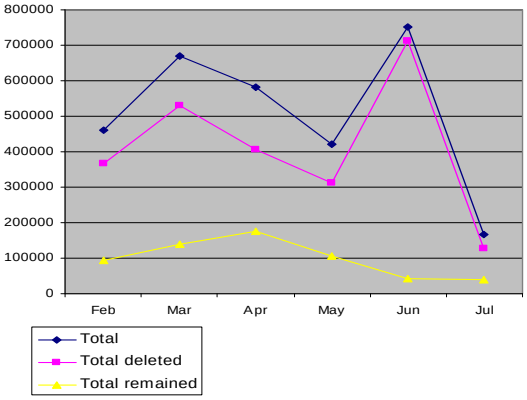

Figure 5: Pre-processed RPL logs per month

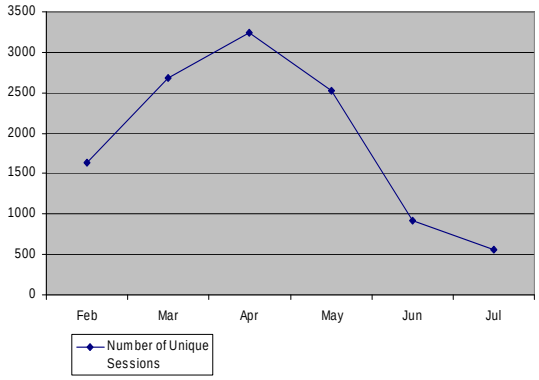

Figure 6: Variation of unique sessions per month

We complete all missing SessionID attributes in the database tables and we used mainly the algorithm below to process this task, then, we determined the number of unique usage sessions using simple SQL queries.

Begin
For each row L_i of T
If L_i[SessionID] is empty
Add_Session(L_i, L_{i-1})
End

Add_Session(L_i, L_j)
Begin
If (L_i[TimeStamp]-L_j[TimeStamp]) < 30)
L_i[SessionID] = L_j[SessionID]
Else
Add_Session(L_i, L_{j-1})
Else
L_i[SessionID] = New_SessionID( )
End

If (L_i[Referrer] = L_j[Url] And
L_i[Agent] = L_j[Agent])

Once we have the session file, we applied, first, CLUTO soft (http://glaros.dtc.umn.edu/gkhome/views/cluto) using a hierarchical partitional clustering algorithm (with cosim similarity and *K*=10 clusters) to cluster learners' sessions. Then, each cluster is represented by a vector composed by pages with highest membership (occurrence). In the second level, association rules are extracted from each cluster, we used Goethals apriori algorithm implementation to compute frequent items, and Goethals association rules implementation to mine AR within each cluster (http://www.adrem.ua.ac.be/~goethals). Component 4 concerns the phase of content modeling. To experiment Nutch crawling process, we built an URL file containing 73469 URLs representing all learning objects available in RPL repository. The Nutch crawler was invoked using the following command line: (`nutch crawl urls -dir crawldir -depth 6`). The crawler uses the URL file for fetching, parsing and indexing the URLs, thus creating an inverted index which will be used to represent the model of educational content. Moreover, in order to make content models more adapted to the pedagogical area, we added LOM attributes to the inverted index. These additional attributes are given by the educational metadata providing descriptions and additional information (author, title, technical requirements, rights management, etc) about learning resources. This information is added automatically to the inverted index thanks to XML files describing files the Standard Defned Learning Objects and Nutch capability to crawl and parse XML files. In component 5, R1 et R2 represent the two main recommendation strategies used to compute what to recommend to the learners. Each recommendation strategy uses as input results returned by Components 1 and 3 and/or Components 2 and 4. Component 5 finally performs the task of delivering recommended links by combining the use of various recommendation approaches or by using them separately based on guidelines given by Component 6 which represents the configuration module specifying a set of entry details to the whole recommender system, such as recommendation strategy and the variation of related parameters (*K*, *W*, *N*, etc). For example, concerning the two-level collaborative filtering approach, to make recommendation, Component 5 starts by classifying the learner's active session under the shape of vector of visited learning objects to the closest cluster (based on Component 1 and 3) using the KNN algorithm (with cosim similarity and *k*=1). Then, matching task between learner's active session and antecedent of AR within the selected cluster is accomplished to produce finally the set of recommended links. Figure 7 shows a learner's activities in RPL platform when browsing courses. Figure 8 shows the recommendations based on Content based Filtering using Nutch. The bottom window contains a list of set-link (shown as "Ensemble i"), each set-link is used to formulate a query whose top search results, as a set, are accessible by clicking the corresponding set-link. These results are shown in the window above. The top-most window contains the actual *content* of a recommended page. Figure 9 shows a list of recommended links based on Collaborative Filtering approach. Notice that for CF recommendations, we also added explanations in the form of the association rules that led to matching inputs. The bottom window of Figure 10 shows list of set-links pointing to a list of recommended links based on cascaded hybrid (collaborative filtering followed by content-based filtering) as shown in the window above. The top-most window contains the actual *content* of a recommended page.

Evaluation

In our experiments, we used the data file obtained after data preprocessing and sessionization, this file contains 11542 unique sessions. We select about 70% of these sessions to compose the training set, the remaining sessions are used for evaluation. Each session *Si* in the evaluation session file is divided into two parts. The first *n LO* references in *Si* are used as the sliding window (the learner's query) to generate recommendations. The remaining portion of *Si* is used to evaluate the generated recommendations. We considered that the maximum size of the sliding window |*w*| is 4, |*w*|<=4. Let *Si(w)* be a set of learning objects composing the active session used by the recommender engine to produce a recommendation set, *t* is a recommendation threshold. The recommender engine takes as inputs *Si(w)* and *t*, *R(Si(w), t)* represents a set of recommended learning objects which score is at least *t*. Let *ESi* a set of remaining learning objects references in a session *Si* within the evaluation session file. The experimental evaluation is based on two metrics: precision and recall defined as following: Precision(*R(Si(w), t)*)=| *R(Si(w), t)* ∩ *ESi*| / | *R(Si(w), t)* | and Recall(*R(Si(w), t)*)=| *R(Si(w), t)* ∩ *ESi*| / |*ESi*|. Precision measure represents the proportion of relevant recommendations to the total number of obtained recommendations, whereas Recall measure or coverage represents the proportion of relevant recommendations to all learning objects that should be recommended (Mobasher et al., 2001). In our experimentation we measured both precision and recall and we varied

recommendation threshold from 0.4 to 1.0. We considered only the recommendation process based on the two-level collaborative approach.

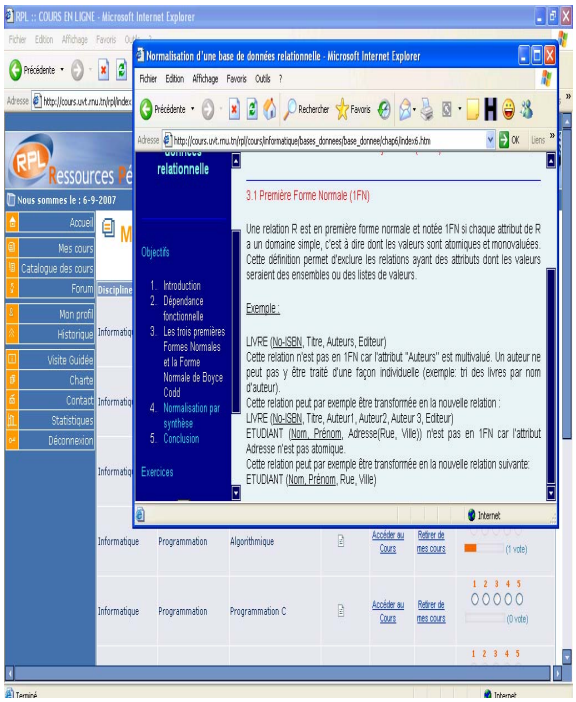

Figure 7: The active learner is browsing material (active user session)

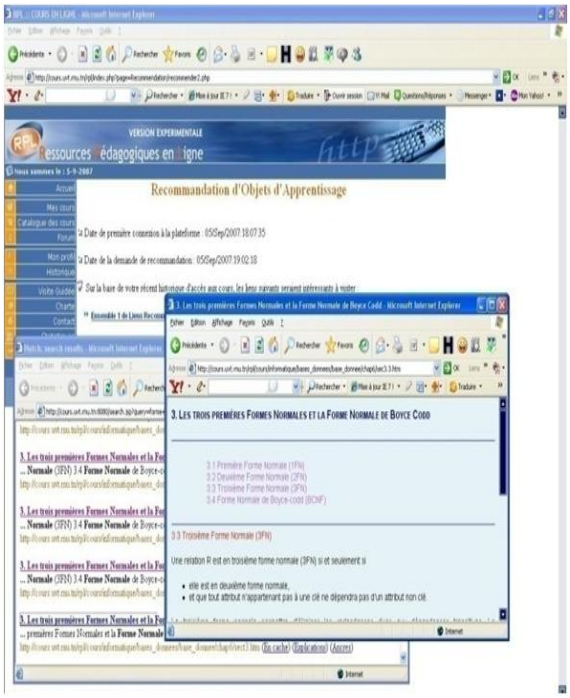

Figure 8: Recommendation using content based filtering approach

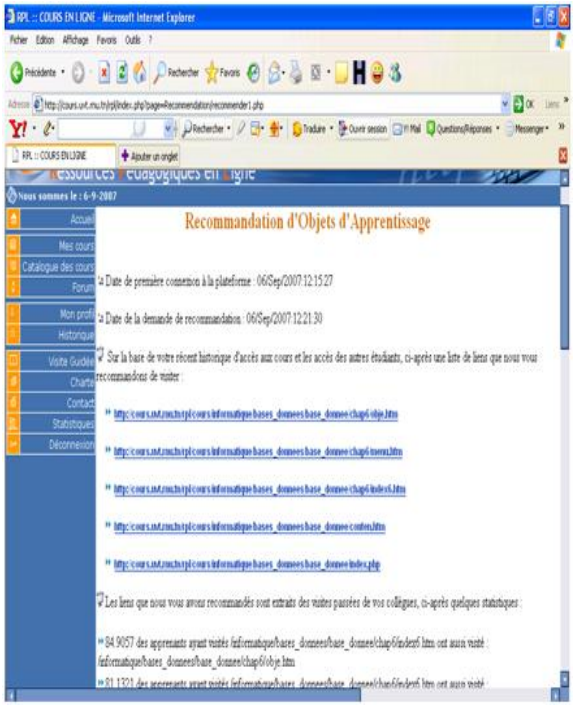

Figure 9: Recommendation (blue links) using collaborative based filtering approach. (Explanations based on collective association rules are listed at the bottom)

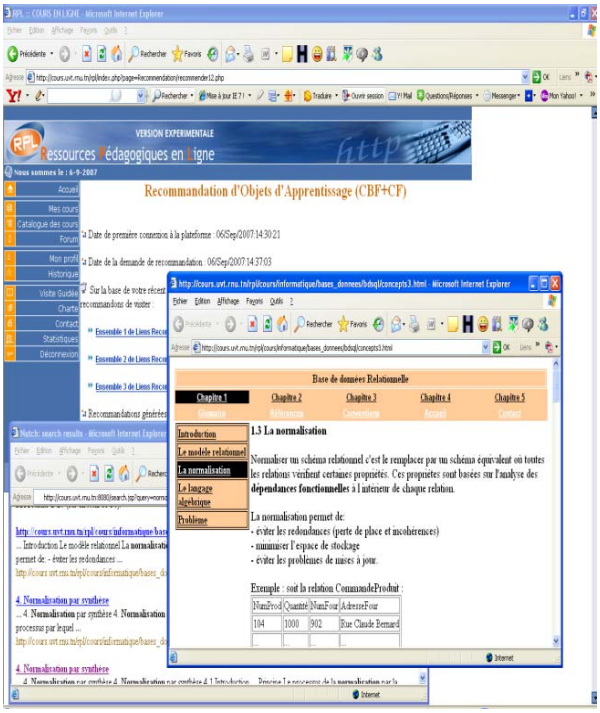

Figure 10: Recommendation using cascaded hybrid (collaborative filtering followed by content-based filtering)

First, we applied CLUTO soft using a hierarchical partitional clustering algorithm (with cosim similarity and *K*=10 clusters) to cluster learners' sessions in the training set. Then, each cluster is represented by a vector composed by pages with highest membership (occurrence). In the second level, association rules are extracted from each cluster, we used Goethals apriori algorithm implementation to compute frequent items, and Goethals association rules implementation to mine AR within each cluster, we considered Confidence=0.4, Support=0.04. In the recommendation phase, we start by classifying the active session *Si(w)* in the closest cluster using the *KNN* algorithm (with cosim similarity and *k*=1). Then, matching task between active session and antecedent of AR within the selected cluster is accomplished to produce finally the set of recommended links. We performed all experiments varying window size from 1 to 4. Finally, for a given recommendation threshold *t*, the mean over all sessions in the evaluation file was computed as the overall evaluation score for each measure. Curves below (Figure 11 and Figure 12) show that precision values increase with the increase of sliding window size and decrease when the number of recommended links increase (conversely for Recall).

```
For each session Si in the Evaluation_session_file i in {1.. p}
    {
    Select the active session Si(w) for recommendation and the Remaining lo set ESi for evaluation
    Select the closest cluster Cj for Si(w)
    For each Rulek in AR file
        {
         If(matching_rulek(Si(w),rulek(cond)))
          {
          Candidate_lo_Si(lo,rank_lo)+=rulek(conseq, rulek(confidence))
          Final_Candidate_lo_Si(lo, rank_lo) = Unique_ Candidate_lo_Si(lo, final_rank_lo)
          with final_rank_lo=sum(over  all rank_lo for the same lo)
          }
    For recommendation threshold t varying from 0.4 to 1.0
        If final_rank_lo > t lo is added Recommended_Set(lo)
         {
          Precision_t+=( Recommended_Set(lo)∩ ESi )/ Recommended_Set(lo)
          Recall_t+=( Recommended_Set(lo)∩ESi)/ESi
         }
    }
For recommendation threshold t varying from 0.4 to 1.0
   {
   Precision_Tot_t =Precision_t/ p
   Recall_Tot_t=Recall_t/p
   }
```

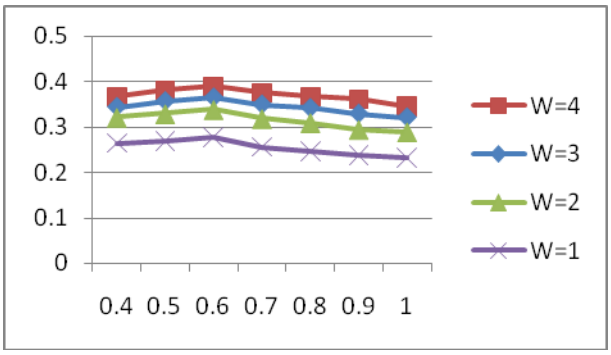

Figure 11: Precision of Recommendations

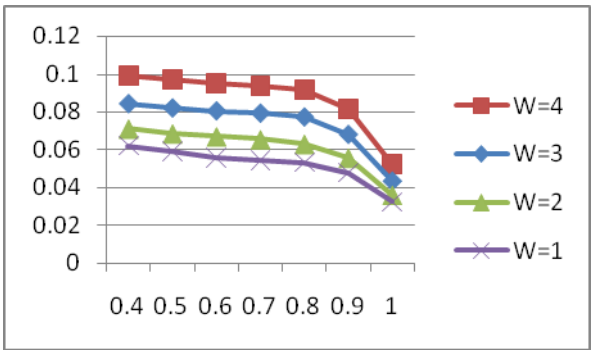

Figure 12: Recall of Recommendations

Conclusions and future work

In this paper, we have outlined the general principles of a new approach to perform personalization in e-learning platforms by resorting to a recommender system relying on web mining techniques and scalable search engine

technology to take care of one of the crucial steps in personalization, which occurs in the "online" phase to compute the recommendations against a possibly massive repository of educational resources in "real time". In the modeling phase, we used Nutch's automated crawling and indexing techniques as well as standardized educational content metadata to build content models, and Web usage mining techniques (clustering and association rule mining) to build learner profiles. Hybrid recommendations (based on CBF and CF) were used in the recommendation phase. We are currently exploring several techniques and strategies in the modeling and recommendation phase in more detail, and performing more evaluations. We are also studying the possibility of integrating educational preferences in the learner's model such as learning styles, media types, etc. The learner's model to consider, in the future work, should be composed of three main components: learner's profile, learner's knowledge and learner's educational preferences. All these components should be detected automatically within e-learning systems. After construction of the student models, we build group models using a three-level collaborative modeling approach. We expect this enrichment of the learner's model to increase the quality of learning object recommendations especially from an instructional point of view.

Acknowledgments

This research was partially supported by National Science Foundation CAREER Award IIS-0133948 to O. Nasraoui.

References

Agrawal, R., & Srikant, R. (1994). Fast Algorithms for Mining Association Rules. *Proceedings of the 20th International Conference on Very Large Databases*, Santiago, Chile.

Brusilovsky, P., & Pesin, L. (1994). An intelligent learning environment for CDS/ISIS users. Proceedings of the interdisciplinary workshop on complex learning in computer environments, Joensuu, Finland.

Brusilovsky, P. (1996). Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction, 6* (2-3), 87-129.

Brusilovsky, P. (2001). Adaptive hypermedia. *User Modeling and User Adapted Interaction, 11* (1/2), 87-110.

Chorfi, H., & Jemni, M. (2004). PERSO: Towards an adaptive e-learning system. *Journal of Interactive Learning Research, 15* (4), 433-447.

Khribi, M. K., Jemni, M., & Nasraoui. O. (2008). Automatic Recommendations for E-Learning Personalization Based on Web Usage Mining Techniques and Information Retrieval. *The 8th IEEE International Conference on Advanced Learning Technologies*, July 1-5, Santander, Spain.

Konstan, J., Miller B., Maltz, D. Herlocker J., Gordon, L. & Riedl, J. (1997). GroupLens: applying collaborative filtering to usenet news. *Communications of the ACM, 40* (3), 77-87.

Meteren, R. V., & Someren, M. V. (2000). Using Content-Based Filtering for Recommendation. *MLnet / ECML2000 Workshop*, May, Barcelona, Spain.

Mladenic, D. (1996). Personal web watcher: Implementation and design. *Technical Report IJS-DP-7472*, Department of Intelligent Systems, J. Stefan Institute, Slovenia.

Mobasher, B., Cooley, R., & Srivastava, J. (2000). Automatic personalization based on Web usage mining. *Communications of the ACM, 43* (8), 142-151.

Mobasher, B., Dai, H., Luo, T., & Nakagawa, M. (2001). Effective Personalization Based on Association Rule Discovery from Web Usage Data. *Proceedings of the 3rd ACM Workshop on Web Information and Data Management*, New York: ACM, 9-15.

Mobasher, B. (2006). Data Mining for Web Personalization. *Lecture Notes in Computer Science, 4321*, 90-135.

Nasraoui, O. (2005). World Wide Web Personalization. In J. Wang (Ed.), *Encyclopedia of Data Mining and Data Warehousing*, Hershey, PA: Idea Group.

Nasraoui, O., Soliman, M., Saka, E., Badia, A., & Germain, R. (2008). A Web Usage Mining Framework for Mining Evolving User Profiles in Dynamic Web Sites. *IEEE Transactions on Knowledge and Data Engineering, 20* (2), 202-215.

Nasraoui, O., Zhang, Z., & Saka, E. (2006). Web Recommender System Implementations in Multiple Flavors: Fast and (Care) Free for All. *Proceedings of the ACM-SIGIR Open Source Information Retrieval*, July, Seattle, WA.

Pazzani M., & Billsus, D. (1997). Learning and revising user profiles: The identification of interesting web sites. *Machine Learning, 27*, 313-331.

Romero, C., & Ventura, S. (2007). Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications, 33* (1), 135-146.

Romero, C., & Ventura, S. (2006). *Data Mining in E-learning*, Southampton: The WITpress.

Romero, C., Ventura, S., Delgado, A. J., & De Bra, P. (2007). Personalized Links Recommendation Based on Data Mining in Adaptive Educational Hypermedia Systems. *Proceedings of the 2nd European Conference on Technology Enhanced Learning*, Crete, Greece, 17-20.

Sarwar, B. M., Konstan, J., Borchers, A., Herlocker, G., Miller, B., & Riedl, J. (1998). Using filtering agents to improve prediction quality in the grouplens research collaborative filtering systems. *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, Seattle, Washington.

Shahabi, C., Zarkesh, A. M., Adibi, J., & Shah, V. (1997). Knowledge Discovery from User's Web-page Navigation. *Paper presented at the 7th IEEE International Conference on Research Issues in Data Engineering*, April 7-8, Birmingham, UK.

Yan, T., Jacobsen, M., Garcia-Molina, H., & Dayal, U. (1996). From user access patterns to dynamic hypertext linking. *Proceedings of the 5th International Worl Wide Web Conference*, Paris, France.

Yao, Y. Y., Hamilton, H. J., & Wang, X. (2002). PagePrompter: An Intelligent Agent for Web Navigation Created Using Data Mining Techniques. *Paper presented at the 3rd International Conference on Rough Sets and Current Trends in Computing*, October 14-16, Malvern, PA, USA.

Zaiane, O. R. (2002). Building a Recommender Agent for e-Learning Systems. *Paper presented at the 7th International Conference on Computers in Education*, December 3-6, Auckland, New Zealand.

Zhao, Y., & Karypis, G. (2005). Hierarchical Clustering Algorithms for Document Datasets. *Data Mining and Knowledge Discovery, 10* (2), 141-168.