Combining Web Usage Mining and Fuzzy Inference for Website Personalization

Olfa Nasraoui and Christopher Petenes

Dept. of Electrical and Computer Engineering
206 Engineering Science Bldg.
The University of Memphis
Memphis, TN 38152-3180
onasraou@memphis.edu
phone number: (901) 678-3251

ABSTRACT

Personalization tailors a user's interaction with the Web information space based on information gathered about them. Declarative user information such as manually entered profiles continue to raise privacy concerns and are neither scalable nor flexible in the face of very active dynamic Web sites and changing user trends and interests. One way to deal with this problem is through a complete automated Web personalization system. Such a system can be based on Web usage mining to discover Web usage profiles, followed by a recommendation system that can respond to the users' individual interests. Significant amounts of error and uncertainty can permeate all the stages of Web personalization. Therefore, we present a fast and intuitive approach to provide Web recommendations using a fuzzy inference engine with rules that are automatically derived from prediscovered user profiles. We perform extensive simulations with real data to study the effect of different parametrization options, and to empirically compare the performance of the proposed approach with that of collaborative filtering and nearest-profile based recommendation strategies. This paper's main contributions are: The proposal of a profile based fuzzy recommendation engine with extensive empirical comparison of different fuzzy input membership derivation and parametrization options, and comparison with approaches based on collaborative filtering and nearest profile recommendations. The proposed fuzzy recommendation method achieves high coverage compared to K-NN and nearest-profile recommendations despite slightly lower precision. Finally, we note that fuzzy recommendations are very intuitive, deal with natural *overlap* in user interests, and are very low in cost compared to collaborative filtering: They are extremely faster and require much lower main memory at recommendation time (no need to store or compare to a large number of instances). This makes fuzzy recommendations suitable for real time recommendations in a *live* setting on today's most active and huge websites.

KEY WORDS: web personalization, web recommendation, web usage mining, fuzzy inference, fuzzy approximate reasoning.

1 Introduction

The flow of information in a completely automated Web personalization system can be prone to significant amounts of error and *uncertainty*. This uncertainty pervades all stages from the user's Web navigation patterns to the final recommendations, including the intermediate stages of logging Web usage, preprocessing, segmenting Web log data into Web user sessions, and learning a usage model from this data. The usage model can come in many forms: from the lazy type modeling used in collaborative filtering, that simply stores all other users' information and then relies on K Nearest Neighbors to provide recommendations from previous history of neighbors or similar users, to a set of frequent itemsets and associations, to a set of clusters of the user sessions, and resulting Web user profiles or summaries. Fuzzy approximate reasoning [12,13] can offer a general framework for the recommendation process. It is this framework that is investigated in this paper. Some previous work in this area has been performed in [7,8,9,14]. Pazzani and Billsus [7] presented a collaborative filtering approach to recommendation, based on users rating specific web pages. Naives Bayes is used as the prediction tool. Kraft et al. [8] construct fuzzy rules from user profiles, and use them to modify queries in information retrieval. This work was based on ratings of web pages whose contents are used to build a user interest profile. Hence, it differs from the approach based on automatically prediscovered Web user profiles that reflect only access behavior and not keywords on a website. Mobasher et al. [9] use pre-discovered association rules and an efficient data structure to provide faster recommendations based on web navigation patterns. Among the most popular methods, the ones based on collaborative filtering and the ones based on fixed support association rule discovery may be the most difficult and expensive to use. This is because, for the case of very *high-dimensional*, *huge*, and extremely *sparse* Web usage data, it is extremely difficult to set a suitable support and confidence threshold to yield reliable and complete web usage patterns. Similarly, collaborative models do not perform well in the face of very sparse data,

and do not scale well to the huge number of users and URLs/items. In association rule based methods, large support thresholds yield only a very small fraction of the total number of itemsets, and smaller support thresholds result in a staggering number of mostly spurious/incorrect URL itemsets. Techniques that are based on clustering to discover profiles are also expected to face serious limitations in the presence of huge, very high-dimensional, and sparse usage data, unless the clustering technique is efficient, robust to noise, and can handle the sparseness. Unsupervised robust multi-resolution clustering techniques [6] can reliably discover most of the Web user profiles/interest groups, because they benefit from the *multi-resolution* mechanism to discover even the less pronounced user profiles, and they benefit from *robustness* to avoid the noisy/spurious profiles that are common with low support thresholds. Their *unsupervised* nature also avoids reliance on input parameters or prior knowledge about the *number* of profiles to be sought. More recent work by Linden et al. [14] used *item-to-item collaborative filtering* as a recommendation strategy in *Amazon.com®*. This approach matches each of the user's purchased and rated items to similar items, then combines those similar items into a recommendation list. A *similar-item table* is built by finding items that customers tend to purchase together.

This paper's contributions are: The proposal of a profile based fuzzy recommendation engine with extensive empirical comparison of different fuzzy input membership derivation and parametrization options and comparison with state of the art approaches based on collaborative filtering (the most popular and widely used) as well as comparison to nearest profile recommendations. The proposed fuzzy recommendation method achieves unprecedented high coverage compared to K-NN and nearest-profile recommendations despite slightly lower precision. Finally, we note that fuzzy recommendations are very intuitive, deal with natural *overlap* in user interests, and very low in cost compared to collaborative filtering: They are extremely faster an require much lower main memory at recommendation time (no need to store or compare to a large number of instances). This makes fuzzy recommendations suitable for real time recommendations in a *live* setting on today's most active and huge websites.

The rest of the paper is organized as follows. In Section 2, we present an overview of profile discovery using Web usage mining. In Section 3, we present the recommendation process based on fuzzy approximate reasoning. In Section 4, we present an empirical evaluation of the recommendation process on real web usage data. Finally, in Section 5, we present our conclusions.

2 Profile Discovery Based On Web Usage Mining With Hunc

The first step in intelligent Web personalization is the automatic identification of user profiles. This constitutes the ***knowledge discovery engine***. These profiles are later used to recommend relevant URLs to old and new anonymous users of a Web site. This constitutes the ***recommendation engine***.

The knowledge discovery part can be executed *offline* by periodically mining *new* contents of the user access log files, and can be summarized in the following steps:

(1) Preprocess log file to extract user *sessions*,
(2) *Categorize* sessions by *Hierarchical Unsupervised Niche Clustering (H-UNC)* [7,8,9],
(3) Summarize the session categories in terms of *user profiles*,
(4) Infer *context-sensitive* URL *associations* from user profiles,

Step 1: Preprocessing the Web Log File to extract User Sessions

The access log of a Web server is a record of all files (URLs) accessed by users on a Web site. Each log entry consists of the following information components: *access time, IP address, URL viewed, …etc*. An example showing two entries is displayed below

17:11:48 141.225.195.29 GET /graphics/horizon.jpg 200
17:11:48 141.225.195.29 GET /people/faculty/nasraoui/index.html 200

The first step in preprocessing [1,2,3] consists of mapping the N_U URLs on a website to distinct indices. A user session consists of accesses originating from the same IP address within a predefined time period. Each URL in the site is assigned a unique number $j \in 1, \ldots, N_U$, where N_U is the total number of valid URLs. Thus, the ith user session is encoded as an N_U-dimensional binary attribute vector $s^{(i)}$ with the property

$$s_j^{(i)} = \begin{cases} 1 & \text{if user accessed } j^{th} \text{ URL} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Step 2: clustering sessions into an optimal number of categories

For this task, we use *Hierarchical Unsupervised Niche Clustering* [6] or *H-UNC***.** *H-UNC* is a *hierarchical* version of a *robust genetic* clustering approach (*UNC*) [5], inspired by nature. A Genetic Algorithm (GA) [14,15,16,17,18] evolves a population of candidate solutions through generations of competition and reproduction until convergence to *one* solution. Hence, the GA cannot maintain population *diversity*. *Niching* methods attempt to maintain a *diverse* population with members distributed among *niches* corresponding to multiple solutions. An initial population of randomly selected sessions is coded into binary chromosome strings that compete based on a density based fitness measure that is highest at the centers of good (dense) clusters. Different niches in the fitness landscape correspond to distinct clusters in the data set. The algorithms are detailed below. Note that the clusters *and their number* are determined automatically. More details on HUNC can be found in [6].

HIERARCHICAL UNSUPERVISED NICHE CLUSTERING ALGORITHM (*H-UNC*):

-Encode binary session vectors
-Set current resolution Level $L = 1$
-Start by applying *UNC* to entire data set w/ small population size;
-Repeat recursively until cluster cardinality or scale become too small {
  -Increment resolution level: $L = L + 1$
  -For each parent cluster found at Level (L-1):
    -Reapply *UNC* [5] only on data subset assigned to this parent cluster
    -Extract more child clusters at higher resolution ($L > 1$)
}

Step 3: Summarizing session clusters into user profiles

After automatically grouping sessions into different clusters, we summarize the session categories in terms of *user profile vectors* [3,4], $\boldsymbol{p}_i$**:** The k^{th} component/weight of this vector ($\boldsymbol{p}_{ik}$) captures the *relevance* of URL_k in the i^{th} profile, as estimated by the conditional probability that URL_k is accessed in a session belonging to the i^{th} cluster (this is the frequency with which URL_k was accessed in the sessions belonging to the i^{th} cluster). The model is further extended to a *robust profile* [3,6] based on robust weights that assign *only* sessions with high robust weight to a cluster's *core*. Unpolluted by noisy (irrelevant) sessions, these profiles give a *cleaner* description of the user interests.

3 The Recommendation Process

Let $U = \{\text{url}_1, \text{url}_2, \ldots, \text{url}_{N_U}\}$ be a set of N_U urls on a given web site visited in web user sessions $\boldsymbol{s}_j$, $j = 1, \ldots, N_s$, as defined in (1). Let $P = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_{N_P}\}$ be the set of N_P Web user profiles computed by the profile discovery engine. Each profile consists of a set of URLs associated with their relevance weights in that profile, and can be viewed as a relevance vector of length N_U, with p_{ik} = relevance of url_k in the i^{th} profile. The problem of recommendation can be stated as follows. Given a current Web user session vector, $\boldsymbol{s}_j = [s_{j1}, s_{j2}, \ldots, s_{jN_U}]$, predict the set of URLs that are most relevant according to the user's interest, and recommend them to the user, usually as a set of *links* dynamically appended to the contents of the Web document returned in response to the most recent Web query. Because the degree of relevance of the URLs that are determined of interest to the user, may vary, it may also be useful to associate the k^{th} recommended URL with a corresponding URL relevance *score*, r_{jk}. Hence it is practical to denote the recommendations for current Web user session, $\boldsymbol{s}_j$, by a vector $\boldsymbol{r}_j = [r_{j1}, r_{j2}, \ldots, r_{jN_U}]$.

3.1 Nearest Profile Based Recommendation Engine

The simplest and most rudimentary approach to Web recommendation is to simply determine the most similar profile to the current session, and to recommend the URLs in this profile, together with their URL relevance weights as URL recommendation scores. Nearest-profile recommendation procedure is intuitively appealing and very simple. In particular, its implementation and deployment in a live setting is very efficient. Essentially, it amounts to a look-up table. However, it has several flaws: ***(i)*** the degree of similarity between the current session and the nearest profile that is identified is not taken into account. While this may not affect the choice of which URLs are recommended, it should affect the recommendation scores, ***(ii)*** the above procedure does not take into account sessions that are similar to more than a single profile, and ***(iii)*** it cannot handle sessions which are different from all known profiles.

We have noticed from our previous work [3,4,6], that not only do profiles overlap in the usage space, but the sessions themselves can be either ***(i)*** of *pure interest*, i.e. clearly identify with a single profile, ***(ii)*** of *mixed interest*, i.e., identify with two or more profiles, even when these profiles do not overlap, or ***(iii)*** of *outlying interest*, i.e., not identifying with any of the pre-discovered strong profiles. The wide spectrum of uncertainties involved in the Web navigation process can be modeled and handled using well studied formal models of uncertainty in fuzzy set theory and soft computing [11,12]. We point, in particular to probabilistic models for case *(i)* and *(ii)*, and general possibilistic models of uncertainty for cases *(ii)* and *(iii)*.

3.2 Fuzzy Approximate Reasoning Based Recommendation Engine

Fuzzy approximate reasoning [11,12] is an inference procedure that derives its conclusions from fuzzy rules and known facts. *Generalized Modus Ponens* is a generalization of *modus-ponens* which is the basic rule of inference in traditional two-valued logic. In two valued logic an implication ($A \rightarrow B$) is used to infer the truth of proposition B from the truth of proposition A, and these truths can only be absolute (i.e., true or false). Generalized Modus Ponens performs logical implication in an *approximate* manner to deal with uncertainties in the facts (input) and in the implication itself. It is formalized as follows:

Fuzzy Rule:	If x is A then y is B
Fact:	x is A'
Conclusion :	y is B'

Where A and B are linguistic variables defined by fuzzy sets on the universes of discourse X and Y, respectively. A fuzzy If-then rule can be defined as a binary fuzzy relation R on the product space $X \times Y$. The relation matrix R, which encodes the fuzzy implication, can be considered as a fuzzy set with a two-dimensional membership function: $\mu_R(x,y) = f(\mu_A(x), \mu_B(y))$ that maps each element in $X \times Y$ to a membership grade in [0,1]. Using the compositional rule of inference, we can formulate the inference procedure in fuzzy reasoning as follows.

$$B' = A' \circ \boldsymbol{R} \qquad (2)$$

where $\circ$ denotes a fuzzy composition operator, consisting of a conjunction, followed by a disjunction. In the context of Web usage based recommendations, X denotes the space of profiles, Y denotes the space of URLs, $\boldsymbol{R}$ is a relation that maps profiles in X to URLs in Y with varying degrees of relevance. The rows of $\boldsymbol{R}$ can be defined by the relevance weights/components of the profile vectors. Input fact A' is a fuzzy set defined on X, thus naturally consisting of the memberships of a given session in each one of the profiles in X. Output B', the composition of A' and $\boldsymbol{R}$ is a fuzzy set defined on the set of URLs, Y. This is the conclusion of the

inferrence procedure, and compared to the original user session that was limited to a crisp set of URLs, *B'* represents the possibility that each URL on the website is of relevance to the current user as inferred via relation matrix ***R***.

We now turn to details of the components of the fuzzy inferrence process: Given current session ***s***, it is desired to infer recommendation ***r***. Hence, the following implication:

$$\boldsymbol{s} \Rightarrow \boldsymbol{r}.$$

Given the Web user profiles discovered by mining the Web logs, the relation ***R*** is defined as follows:

$$R_{ik} = p_{ik}$$

The input fuzzy set is derived from the current session $x = \boldsymbol{s}$ by computing the similarity value between ***s*** defined in (1) and each profile, $\boldsymbol{p}_i$, as follows:

$$\mu_{\boldsymbol{si}} = \mu_{\boldsymbol{s}}(i) = sim\ (\boldsymbol{s}, \boldsymbol{p}_i) \qquad (3)$$

where *sim()* is given by (7), (8), or any other appropriate normalized similarity measure. The inferrence procedure concludes the recommendation ***r*** as the possibility for URL relevance via the following composition

$$\mu_{\boldsymbol{rk}} = \mu_{\boldsymbol{r}}(\mathrm{URL}_k) = \mu_{\boldsymbol{s}}(i) \circ \boldsymbol{R} = \bigvee_i [\wedge (\mu_s(i), R_{ik})], \qquad (4)$$

where $\vee$ and $\wedge$ can be replaced by any appropriate t-conorm/union and t-norm/intersection, respectively. The membership levels of the k^{th} URL in the final recommendation set are $\mu_{\boldsymbol{rk}}$, and these are considered as the recommendation *scores*. Below is a description of the algorithm.

Fuzzy Approximate Reasoning Based Recommendation Engine:

Input: a set of N_P *profiles* and a *current* session ***s***

Output: *recommendation* vector ***r***.

Pseudo code:

- Generate N_P–component fuzzy session profile vector $\mu_{\boldsymbol{s}}$ from current session using appropriate similarity measure,
- Compose $\mu_{\boldsymbol{s}}$ with fuzzy profile relation matrix to get N_U–component recommendation vector $\boldsymbol{r} = \mu_{\boldsymbol{r}}$,
- Present top N_R URLs with non zero $\mu_{\boldsymbol{r}}$ sorted in descending order of $\mu_{\boldsymbol{r}}$, not including URLs in current session.

3.2.1 Recommendation Strategy Options

Since the evaluation of web recommendations is very subjective, one should investigate several alternatives for performing the fuzzy inferrence process. This involves, varying: ***(i)*** the way the *input* membership, $\mu_{\boldsymbol{si}}$, are defined (probablistic, possibilistic, normalized possibilistic), ***(ii)*** the type of profiles used to generate input memberships $\mu_{\boldsymbol{si}}$, (raw profiles: $\boldsymbol{p}_i$,, or crisp α-cuts: $\boldsymbol{p}^{\alpha}{}_i$,) ***(iii)*** the type of *t-norm/intersection* used in the composition (min, product), ***(vi)*** the type of *t-conorm/union* used in the composition (max, bounded-sum), and ***(v)*** the derivation of the *relation matrix* ***R*** from the profile vectors, $\boldsymbol{p}_i$, (original profiles, thresholded fuzzy, defuzzified α-cuts), and ***(vi)*** the *final recommendation scores* derived from $\mu_{\boldsymbol{r}}(\mathrm{URL}_k)$ (unmodified composition outputs, thresholded fuzzy, defuzzified α-cuts). These variations are summarized below.

3.2.2 Input Membership

Probabilistic-type memberships are not necessarily derived from assumptions about a probablity distribution, but share in common with probablities, the fact that they sum to 1, i.e.,

$$\sum_{i-1}^{N_P} \mu_{si} = 1$$

This is accomplished by normalizing the memberships as follows:

$$\mu_{si_{norm}} = \mu_{si} \Big/ \sum_{i-1}^{N_P} \mu_{si} \qquad (5)$$

Possibilistic memberships are the original similarity values to the profiles, since these are typicalities.

The *normalized possibilistic* memberships have a maximum value of 1, and are obtained as follows

$$\mu_{si\,pos\_norm} = \mu_{si} \Big/ \max_i \mu_{si} \qquad (6)$$

The individual membership, $\mu_{\boldsymbol{si}}$, of an input session, ***s***, in the i^{th} profile, $\boldsymbol{p}_i$, is computed using the similarity score between this session and the profile. For example, the cosine similarity can be used to yield the following input membership,

$$\mu_{si}^{cosine} = \frac{\sum_{k=1}^{N_U} p_{ik} s_k}{\sqrt{\sum_{k=1}^{N_U} p_{ik} \sum_{k=1}^{N_U} s_k}} \qquad (7)$$

If a hierarchical Web site structure should be taken into account, then a modification of the cosine similarity, introduced in [3,4], that can take into account the Website structure can be used to yield the following input membership,

$$\mu_{si}^{web} = \max\left\{ \frac{\sum_{l=1}^{N_U}\sum_{k=1}^{N_U} p_{il} S_u(l,k) s_k}{\sum_{k=1}^{N_U} p_{ik} \sum_{k=1}^{N_U} s_k}, \mu_{si}^{cosine} \right\} \qquad (8)$$

where $\boldsymbol{S_u}$ is a URL to URL similarity matrix that is computed based on the amount of overlap between the paths leading from the root of the website (main page) to any two URLs, and is given by

$$S_u(i,j) = \min\left(1, \frac{|p_i \cap p_j|}{\max(1, \max(|p_i|, |p_j|) - 1)}\right)$$

We refer to this special Web session similarity.

3.2.3 Profile vectors used to Compute Similarities in (7) and (8)

The similarity measure can use the profile vectors or use binary thresholded α-cuts of the profiles if these are viewed as fuzzy sets over the domain of URLs, i.e.,

$$p_{ik}^{\alpha} = \begin{cases} 1, if\ \ p_{ik} \geq \alpha \\ 0, otherwise \end{cases} \qquad (9)$$

binary profiles require a simple binary similarity (based on matches) while raw (real) profiles require real counterparts of the similarity measures.

3.2.4 Recommendation Strategy Based on Fuzzy Approximate Reasoning

After mining the profiles in the Web usage mining stage, the fuzzy profile-to-URL relation matrix is derived as described in Section 3.2. Suppose that a current user visits l_s URLs on the same website. Here, l_s denotes the length of the current session. The goal of the recommendation process is to generate dynamic recommendations given the user's access pattern so far. The recommendations are in the form of a set of URLs and their corresponding recommendation scores as given in (4), and can also be viewed as a fuzzy set over the URL domain.

3.2.5 Evaluating the recommendation process

Each actual completed session, $\boldsymbol{s}_T$, is treated as ground-truth, a subset of this session is treated as incomplete current sub-session, $\boldsymbol{s}_j$, and the fuzzy recommendations are treated as predicted complete session. This process is very similar to an information retrieval problem. Hence the evaluation proceeds by computing the *precision* and the *coverage* of the recommendations.

Let $r^*_j = r_j - \boldsymbol{s}_j$. This corresponds to the recommendations obtained after omitting all URLs that are part of the current subsession. Also, let $\boldsymbol{s}^*_j = \boldsymbol{s}_T - \boldsymbol{s}_j$, be the set of ground truth URLs, not including the ones in the current subsession being processed for recommendations. The recommendation precision is given by

$$prec_j = \frac{\sum_k^{N_U} r^*_{jk} s^*_{jk}}{\sum_k^{N_U} \left(r^*_{jk}\right)^2} \tag{12}$$

and coverage is given by

$$\mathrm{cov}_j = \frac{\sum_k^{N_U} r^*_{jk} s^*_{jk}}{\sum_k^{N_U} \left(s^*_{jk}\right)^2} \tag{13}$$

4 Experimental Results

4.1 Mining The User Profiles From Anonymous Web Usage Data

Hierarchical Unsupervised Niche Clustering (H-UNC) [6] was applied on a set of web sessions preprocessed from the 12 day access log data of the Web site of the department of Computer Engineering and Computer Sciences at the University of Missouri, Columbia. After filtering out irrelevant entries, the data was segmented into 1703 sessions accessing 343 distinct URLs. The maximum elapsed time between two consecutive accesses in the same session was set to 45 minutes. We applied H-UNC to the Web sessions using a maximal number of levels, L = 5, in the hierarchy, and the following parameters that control the final resolution: N_{split} = 30, and σ_{split} = 0.1. H-UNC partitioned the Web users sessions into 20 clusters at level 5, and each cluster was characterized by one of the profile vectors, $\boldsymbol{p}_i$, i=0, ...,19. Some of these profiles are summarized in Table 1.

TABLE 1. SOME OF THE 20 DISCOVERED PROFILES

No.	size	Relevant URLs/Profile	Profile Description
0	106	{0.29 - /staff.html} {0.92 - /} {0.98 - /people.html} {0.99 - /people_index.html} {0.97 - /faculty.html} {0.15 - /degrees.html} {0.20 - /research.html} {0.35 - /grad_people.html} {0.26 - /undergrad_people.html}	Main page, people, faculty, staff, degrees, research.
1	104	{0.99 - /} {1.00 - /cecs_computer.class}	Main page only
3	61	{0.80 - /} {0.48 - /degrees.html} {0.23 - /degrees_grad.html} {0.23 - /degrees_grad_index.html} {0.23 - /deg_grad_genor.html}	Main page and *graduate* degrees
4	58	{0.72 - /} {0.97 - /degrees_undergrad.html} {0.95 - /degrees_index.html} {0.97 - /bsce.html} {0.52 - /bscs.html} {0.34 - /bacs.html} {0.34 - /courses.html} {0.34-/courses100.html} {0.26 - /general.html} {0.22- /courses300.html} {0.81 - /degrees.html} {0.19 - /courses200.html}	Main page, *undergraduate* courses, degrees, especially *undergraduate* degrees.
13	38	{0.47 - /~shi} {0.82 - /~shi/cecs345} {0.21 - /~shi/cecs345/java_examples} {0.34 - /~shi/cecs345/references.html}	CECS345 Java examples/references

4.2 Recommendations Based On Fuzzy Approximate Reasoning

4.2.1 Fuzzy Recommendation Simulations

Given the prediscovered profiles, and a set of Web sessions extracted from the same Web log file, we treat every complete session as ground-truth session. For each such ground-truth session, all possible subsets of this session consisting of between 1 and 9 URLs are considered as current test subsessions, for which the recommendations are computed, and the coverage and precision measures are averaged for each subsession size, separately, and plotted on Figures 2 and 3, respectively. This procedure generated roughly 380,000 separate recommendation tests, each tested using 16 different recommendation scenarios, corresponding to different combinations of input membership generation, composition, ...etc, as described in Sec. 3.2.1. For instance, with regard to item ***(i)***, two types of similarities were used: cosine or Web session (denoted as Cosine and WS respectively in the plots). With regard to items ***(iii)*** and ***(iv)*** in Sec. 3.2.1, two different compositions were tested: Max-Min and Bounded Sum-Min (denoted as MM and BS in the plots). with regard to items ***(ii)***, two different types of profiles were tested: raw profiles generate real similarities (denoted as Real Cosine and Real WS in plots), and crisp α–cuts with α=0.2 (denoted as Binary Cosine - .2 Thresholded Profile or Binary WS - .2 Thresholded Profile in plots, and was only tested for max-min composition). Finally , with regard to item ***(vi)*** in Sec. 3.2.1, once similarities are computed, either raw fuzzy input session memberships were used, or crisp α –cuts with α=0.001, 0.2, and 0.3 (denoted Type of Input Similarity - α Thresholded To Bin in plots).

While it is difficult to make absolute conclusions about all combinations, it seems clear that max-min composition exceeds bounded-sum composition both with respect to better coverage and better precision. It is also clear that cosine similarity is favorable from a precision point of view. However, Web session (WS) similarity exceeds the cosine similarity in coverage. This is because Web session similarity [6] takes into account the site structure to capture similarity even between distinct URLs that are close to each

other in the web site hierarchy. This tends to improve coverage, but only at the risk of slightly less precision. Also, using crisp profile α –cuts to binarize input session memberships after computing the similarities resulted in improved precision, but at the expense of worse coverage. The effect of thresholding the input memberships using a higher α –cut, dramatically improved precision, but also resulted in significamtly worse coverage. The results do confirm to what is expected in terms of the natural trade-off between precision and coverage in recommendation systems. Thresholding the profiles may improve precision because of the removal of weak URLs in the membership calculations, while enhancing strong URLs, thus acting like a filter that reduces noise from the input session membership vector. While it is difficult to make any definitive conclusions from our simulations, it is reasonable to conclude that using crisp profile α –cut of 0.2 to compute input memberships via the cosine similarity resulted in the best overall tradeoff between precision and coverage. Most other combinations of options tend to result in significant improvements in coverage, but only at the expense of significant deterioration in precision, or vice-versa. The plots showing a sharp drop towards 0 after a specific session size are a result of the absence of any longer sessions in some profiles. Coverage improves greatly with session size, while precision decreases. Extremely low values point to the large number of noisy sessions that have not been removed from the data set.

4.2.2 Comparison with Nearest Profile and Collaborative Filtering Based Recommendations

The Nearest Profile based recommendation approach is based on simply using the closest profile based on a suitable similarity measure (Web session or cosine) as explained in Section 3.1. The Collaborative filtering approach is based on using K Nearest Neighbors (K-NN) followed by top-N recommendations for different values of K and N. First the closest K complete sessions from the entire history of accesses are found. Then the URLs present in these top K sessions are sorted in decreasing order of their frequency, and the top N URLs are treated as the recommendation set. Figure 5 shows how the coverage is much lower for both of these strategies compared to the fuzzy recommendation approach. This is expected since both approaches base their predictions on a closest identified *local* neighborhood of instances (for K-NN) or on a model based on profiles (for Nearest Profile). Both approaches are limited when it comes to sessions that are on the overlap between different profiles as opposed to the fuzzy approach that naturally interpolates the effect of several profiles, and hence achieve a more global viewpoint to allow better coverage. While having lower coverage, the nearest profile and K-NN approach achieve better precision again because of their more local nature. This results in the average F1 measure shown in Figure 4. In this figure, we see that fuzzy recommendations are superior to crisp nearest profile based recommendations, especially at higher subsession sizes, while K-NN collaborative filtering achieves the best tradeoff bewteen coverage and precision for small subsession sizes, and is then topped by fuzzy recommendations. We explain the superior performance of fuzzy recommendations for longer test sessions to the fact that longer sessions become more likely to have originated from more than one profile, and this is exactly where fuzzy recommendations are expected to be superior.

The different behavior of recommendation strategies for different session lengths suggests that it may be beneficial to combine different recommenders depending on the session length. This will help take advantage of the strengths of all the recommendation strategies. In particular, we also note that even in the fuzzy recommendation strategies

From the point of view of cost or time and memory complexity, it is interesting to note that offline training takes longer with the profile based approach (both fuzzy and crisp). However this can be done on a back end computer and not on the server, and is therefore an offline process that does not affect the operation of the web server. On the other hand, both fuzzy and nearest profile based approaches are extremely fast at recommendation time and require a minimal amount of main memory to function since they work with a mere summary of the previous usage history instead of the entire history as in collaborative filtering. In our simulations, fuzzy recommendations with non-optimized Perl script (non-compiled) code running on a 2 GHz Pentium 4 Linux PC operated at an average of 48 recommendations per second[1]. The far more computationally complex K-Nearest Neighbor generated recommendations at a leisurely 2 recommendations per second.

5 Conclusions

The flow of information in a Web personalization system can be prone to significant amounts of error and uncertainty. This uncertainty pervades all stages from the user's Web navigation patterns to the final recommendations. Fuzzy approximate reasoning seems to be a natural framework for the recommendation process. Taking *current* Web *usage*/navigation information or *context* into account captures a challenging aspect that is recognized as very influential in Web information retrieval/search engines. We presented a simple, intuitive, and fast approach to provide dynamic predictions in the Web navigation space. Real noisy Web usage data was used as a simulation testbed for the fuzzy recommendation system. The proposed approach is efficient since only prediscovered profiles (offline) need to be compared. We took advantage of the sparsity of Web usage data to enable a direct storage and access to the relation matrix's columns by hashing. The proposed approach is fit for *real-time* recommendations (on average less than 0.02 secs. per recommendation on a 2 GHz Pentium 4 Linux PC). In addition to being fast, the proposed fuzzy recommendation method achieves unprecedented high coverage compared to K-NN and nearest-profile recommendations despite slightly lower precision. Finally, we note that fuzzy recommendations are very intuitive, deal with natural *overlap* in user interests. This makes fuzzy recommendations suitable for real time recommendations in a *live* setting on today's most active and huge websites. We are currently performing more offline and online tests using various recommendation strategies.

[1] Here, 1 *recommendation per second* is actually a full set of recommendations for a given session and *not* individual URLs per second.

References

[1] M. Perkowitz and O. Etzioni. Adaptive web sites: Automatically learning for user access pattern. Proc. 6th int. WWW conference, 1997.

[2] R. Cooley, B. Mobasher, and J. Srivastava, Web Mining: Information and Pattern discovery on the World Wide Web, Proc. IEEE Intl. Conf. Tools with AI, Newport Beach, CA, pp. 558-567, 1997.

[3] O. Nasraoui and R. Krishnapuram, and A. Joshi. Mining Web Access Logs Using a Relational Clustering Algorithm Based on a Robust Estimator, 8th International World Wide Web Conference, Toronto, pp. 40-41, 1999.

[4] O. Nasraoui, R. Krishnapuram, H. Frigui, and A. Joshi. Extracting Web User Profiles Using Relational Competitive Fuzzy Clustering, International Journal on Artificial Intelligence Tools, Vol. 9, No. 4, pp. 509-526, 2000.

[5] O. Nasraoui, and R. Krishnapuram, "A Novel Approach to Unsupervised Robust Clustering using Genetic Niching," Proc. of the 9th IEEE International Conf. on Fuzzy Systems, San Antonio, TX, May 2000, pp. 170-175.

[6] O. Nasraoui and R. Krishnapuram. "A New Evolutionary Approach to Web Usage and Context Sensitive Associations Mining," International Journal on Computational Intelligence and Applications - Special Issue on Internet Intelligent Systems, Vol. 2, No. 3, pp. 339-348, Sep. 2002.

[7] M. Pazzani and D. Billsus, "Learning and revising User Profiles: The identification of Interesting Web Sites," Machine Learning, Arlington, 27, pp. 313-331, 1997.

[8] Kraft, D.H., Chen, J., Martin-Bautista, M.J., and Vila, M.A., "Textual Information Retrieval with User Profiles Using Fuzzy Clustering and Inferencing," in Szczepaniak, P.S., Segovia, J., Kacprzyk, J., and Zadeh, L.A. (eds.), Intelligent Exploration of the Web, Heidelberg, Germany: Physica-Verlag, 2002.

[9] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa, "Effective personalizaton based on association rule discovery from Web usage data," ACM Workshop on Web information and data management, Atlanta, GA, Nov. 2001.

[10] J. H. Holland. *Adaptation in natural and artificial systems*. MIT Press, 1975.

[11] L. Zadeh (1965). Fuzzy sets. Inf. Control 8, 338-353.

[12] Klir, G. J., and Yuan, B., Fuzzy Sets and Fuzzy Logic, Prentice Hall, 1995, ISBN 0-13-101171-5.

[13] R. Agrawal and R. Srikant (1994), Fast algorithms for mining association rules, Proceedings of the 20th VLDB Conference, Santiago, Chile, pp. 487-499.

[14] G. Linden, B. Smith, and J. York, "*Amazon.com* Recommendations Iem-to-item collaborative filtering", IEEE Internet Computing, Vo. 7, No. 1, pp. 76-8

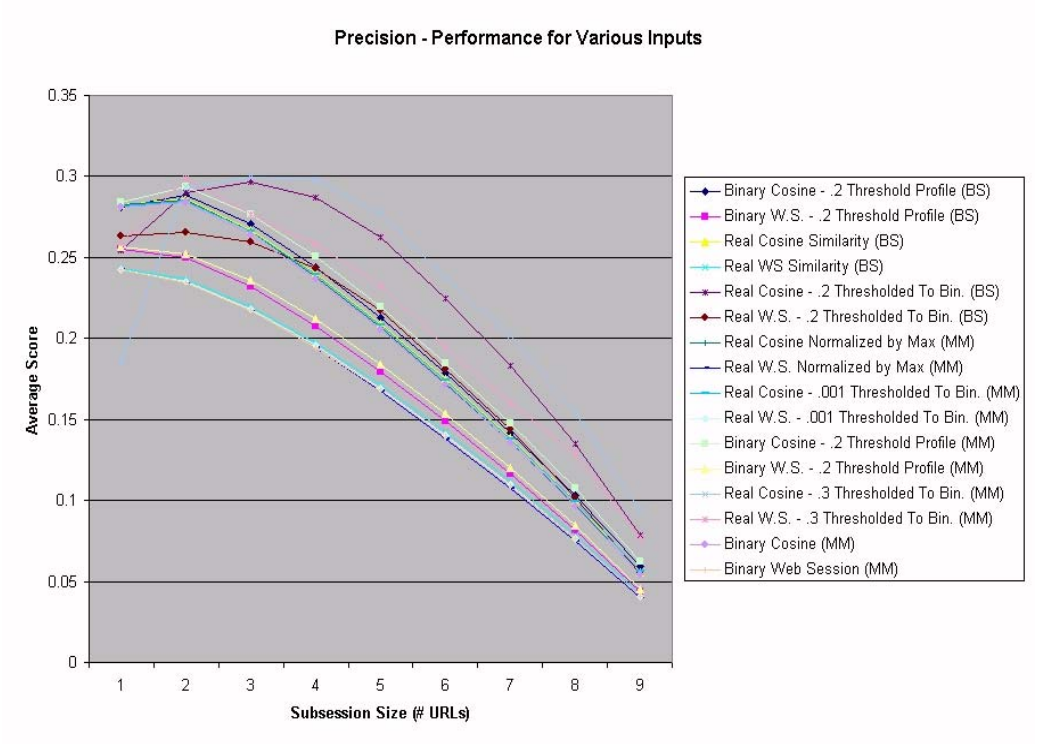

Figure 1: Average Precision per subsession size for different fuzzy strategies with different fuzzy input memberships derived from Cosine or Web session similarity: Min-Max (MM), Min-Bounded Sum (BS), and different fuzzy membership vector normalizations: Normalization by Maximum (N-Max), different α-cut to binarize profiles: α=.2 (.2 Thresholded Profile) before computing input memberships with binary cosine or Web Session similarity (Binary Cosine or Binary W.S.), and different α-cut to binarize resulting final input memberships (α Thresholded to Bin) before the fuzzy composition

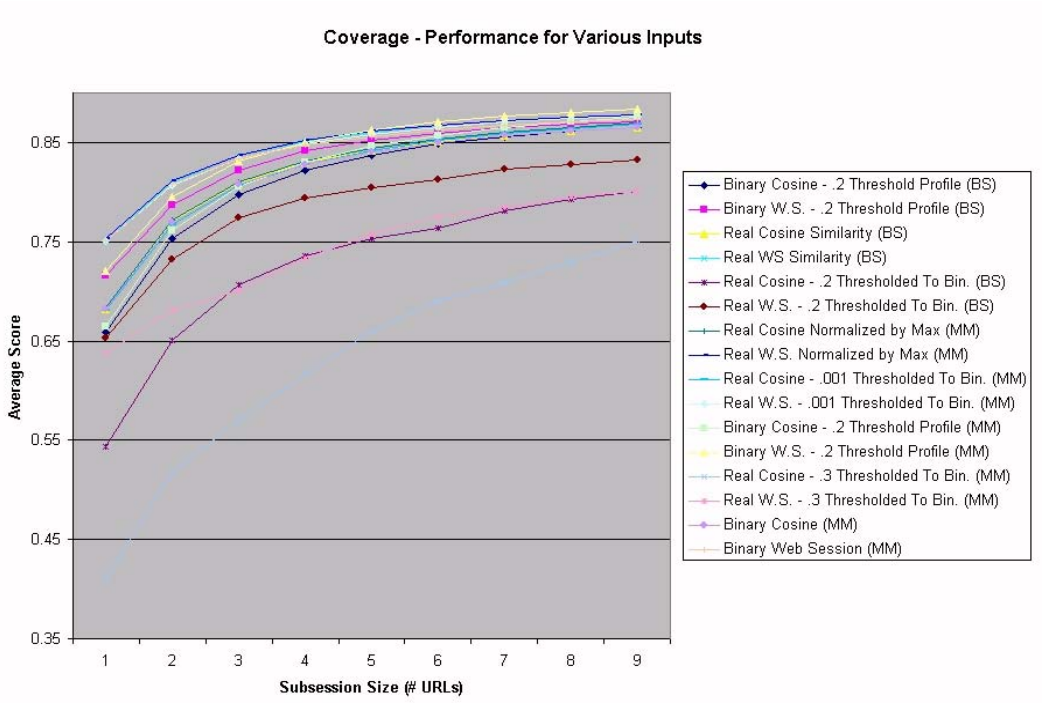

Figure 2: Average coverage per subsession size for different fuzzy strategies with different fuzzy input memberships derived from Cosine or Web session similarity: Min-Max (MM), Min-Bounded Sum (BS), and different fuzzy membership vector normalizations: Normalization by Maximum (N-Max), different α-cut to binarize profiles: α=.2 (.2 Thresholded Profile) before computing input memberships with binary cosine or Web Session similarity (Binary Cosine or Binary W.S.), and different α-cut to binarize resulting final input memberships (αThresholded to Bin) before the fuzzy composition

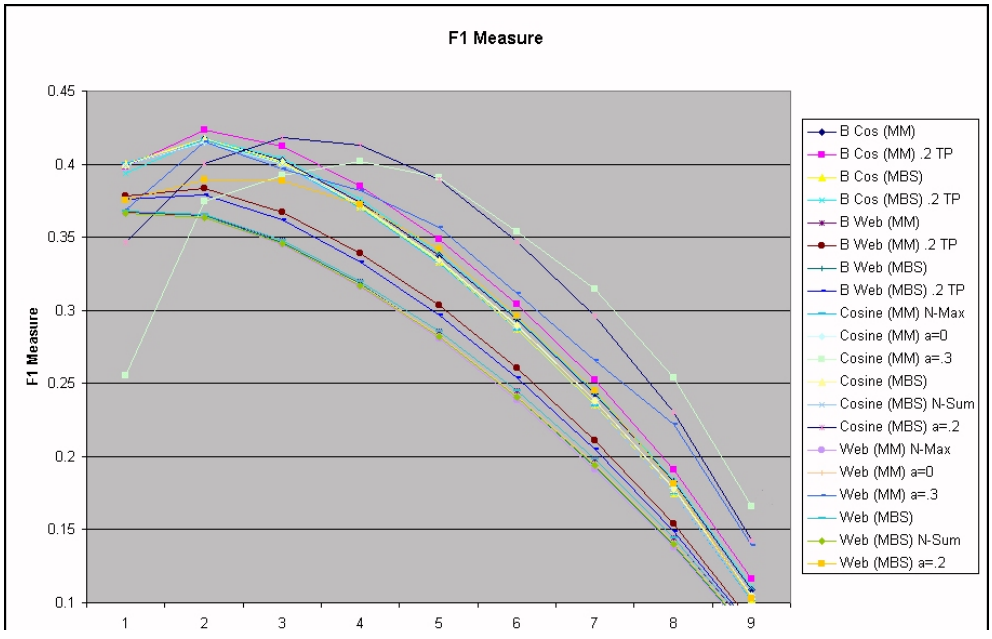

Figure 3: Average F1 Measure per subsession size for different fuzzy strategies with different fuzzy input memberships derived from Cosine or Web session similarity: Min-Max(MM), Min-Bounded Sum(MBS), and different fuzzy membership vector normalizations: Normalization by Maximum (N-Max) and Normalization by Sum (N-Sum), different α-cut to binarize profiles: α=0 or α=.2 (.2 TP) before computing input memberships with binary similarity (B Cos or B Web), and different a-cut to binarize resulting final input memberships

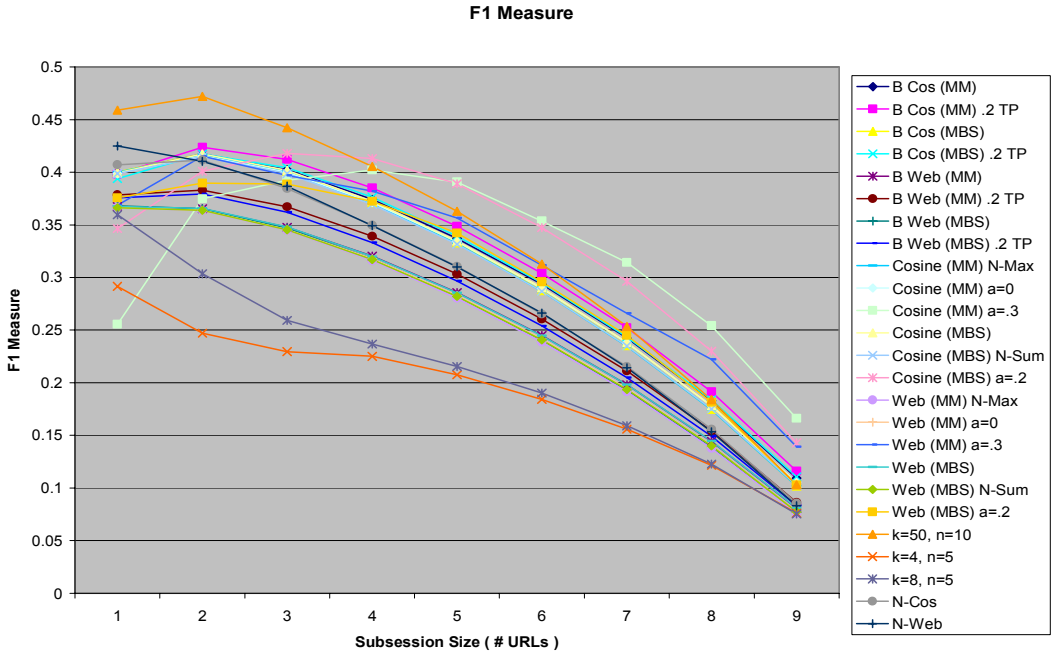

Figure 4: Average F1 Measure per subsession size for different fuzzy strategies with different fuzzy input memberships derived from Cosine or Web session similarity: (Min-Max(MM), Min-Bounded Sum(MBS), and different fuzzy membership vector normalizations: Normalization by Maximum (N-Max) and Normalization by Sum (N-Sum), different α-cut to binarize profiles: α=0 or α=.2 (.2 TP) before computing input memberships with binary similarity, and different a-cut to binarize resulting final input memberships; Nearest Profile based on Cosine (N-Cos) and Web session (N-Web) similarities; and Collaborative Filtering with k-Nearest Neighbor strategy and top-n recommendations

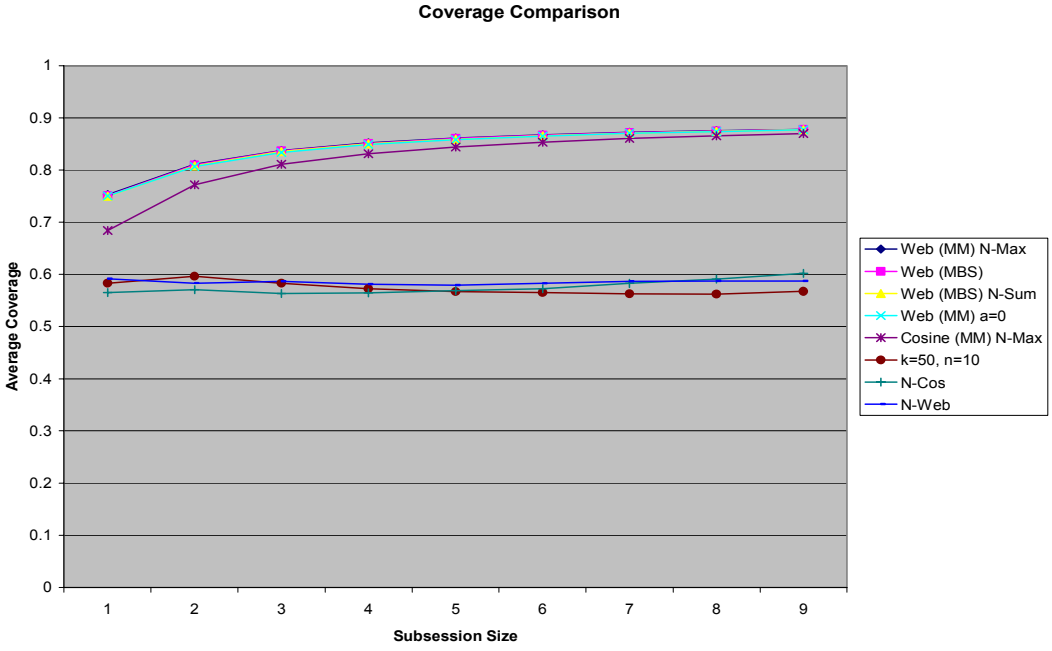

Figure 5: Comparison of Average Recommendation Coverage per subsession size for best four fuzzy strategies with different fuzzy input memberships derived from Web session similarity: (Min-Max(MM), Min-Bounded Sum(MBS), and different fuzzy membership vector normalizations: Normalization by Maximum (N-Max) and Normalization by Sum (N-Sum); Nearest Profile based on Cosine (N-Cos) and Web session (N-Web) similarities; and Collaborative Filtering with K-Nearest Neighbor strategy and top-n recommendations (k=50, n=10)