# A Fuzzy Logic Approach for Dynamic User Interests Profiling

Abd El Heq Silem, Hajer Taktak, and Faouzi Moussa
Faculty of Sciences of Tunis, LIPAH LR11ES14
University of Tunis El Manar, El manar1
Tunis, Tunisia
Email: {hakou.silem, taktakhajer, faouzimoussa}@gmail.com

***Abstract*— The user profile is the virtual representation of the user that holds a variety of user information such as personal data, interests, preferences, and environment. In literature, there are two different techniques for profiling user interests. The first one is based on the retrieval of text from the user browsing history; this technique has a high probability to generate a false interest from uninteresting websites. The second technique is based on user behavior (factors like scrolling speed or time spent) and navigation history. The proposed approach using the second technique does not use enough factors and calculates the weight of each factor via predefined ranges, which is not accurate for all users. This technique generates incorrect factor weight and false user interests. In this paper, we propose an approach that employs Fuzzy Logic with several factors (scrolling speed, time spent, and the number of visits) to automatically build and update the user profile from the user's browsing history. The target websites for this approach are websites that contain text content rather than visual content. This approach adapts the range of each factor according to the user habits using Fuzzy Logic, which improves accuracy and avoids a predefined factor range. Finally, we use an ontology-based model to store the user profile.**

***Keywords-Context-Awareness; Fuzzy Logic; Fuzzy Logic System; User Profiling; User Behavior.***

## I. INTRODUCTION

Personalization systems are very important in computer science due to their ability to provide relevant content to the user and due to the growth of accessible information. The personalization system must act according to user preferences and interests (in other words, to provide content relevant to the user). To solve this problem, it is necessary to collect and store user personal information, preferences, and interests. This is called user profiling.

The user profiling process has two significant challenges. The first challenge is the creation of the user profile, called the cold start (the system has no information about the user to be used in the personalization). The second challenge is to keep the existing information in the profile up to date according to the user changing preferences. In literature, there are three main approaches [1][2] about user profile information collection:

- Explicit approach (static profiling): This approach collects data directly from the user using forms or surveys, which generate a very accurate profile at the beginning. This accuracy deteriorates over time, especially when the user does not fill in the new surveys.
- Implicit approach (dynamic profiling): this approach infers information about the user without the user's intervention, based on the browsing history and behavior. The problem in this approach is the cold start and the accuracy of inferred information about the user.
- Hybrid approach: combines the previous approaches to override their weaknesses and increase their benefits. It creates the profile of users using the explicit approach. Then, it maintains the profile updated using the implicit one.

The rest of the paper is organized as follows: In Section 2, we discuss some of the related works. Section 3 presents the Fuzzy Logic system. In Section 4, we discuss the user profile model, and Section 5 concludes the paper.

## II. RELATED WORK

In literature, there are three main user profiling methods: the content-based, the collaborative, and the hybrid method [2]. The content-based methods create the user profile according to the user’s behavior (detect interest from the behavior). Then, they select content with a strong correlation to the created profile. The collaborative methods are based on a similar rating of users. These methods create a profile for a group of users who have the same rating or similar taste and make a recommendation based on the group rating. The hybrid techniques combine the two previous methods to improve the strengths and overcome the weaknesses of each method.

Tchantchou et al. [3] propose a multi-agent architecture for user interest profiling and an improved algorithm for mapping the Conceptual Clustering Concept (ICCC). The user profile contains both explicit and implicit interests. Implicit interests are derived from the user browsing history using the ICCC algorithm. The architecture extracts the text from the visited webpages, removes stop words, and reduces each word to its stem. Then, it assigns weights to those stems according to the stem position and occurrence and creates the term vector of each website. After that, the architecture will map each website to a concept based on the ICCC algorithm and an ontology that contains a set of concepts and websites. Finally, it updates the profile of users with the new weights. This architecture does not use user behavior to detect user interest. It only uses the text extracted from visited webpages,

 ISBN: 978-1-61208-761-0 

which does not differentiate between interesting and uninteresting websites and it will generate false interests (if the user visits a set of random uninteresting websites in a session).

Moawad et al. [4] propose a multi-agent architecture for customization of Web search according to the user profile. The profile is built from the user's explicit information and interests (collected explicitly). The architecture implicitly updates the profile by capturing the interaction and the browsing history of the user. First, it retrieves the stems from webpages like the precedent work [3] (the authors add user action, such as copying and bookmark, to calculate the weight). Using the Wordnet ontology [5] (Wordnet ontology is a lexical database of English), the architecture recovers the first common topic between all the stems (of the same webpage) and creates a triplet (stem, topic, and weight of stem). Finally, the topic weight is calculated based on the weight of all its stems and the number of stems. In this work, the authors rely only on two actions to determinate the interesting webpages. Bookmarking or copying text from a webpage does not always mean that the user is interested in this type of content and vice versa (in many cases, users do not bookmark or copy text from interesting webpages). Therefore, the results of this technique are misleading and far from being reliable.

Singh et al. [6] propose a multi-agent architecture for the dynamic construction of the user profile according to user's browsing history, the scrolling speed, time spent, and user behavior at the desktop (such as applications and files opened). The architecture is a client/server architecture where the client-side is responsible for collecting user information (desktop and browsing behavior). It analyses that information to create the user profile (estimation of user interests). The server-side maintains and updates the profiles of all users provided by the client-side. Then, it groups these users according to their interests and provides content based on these groups. In this work, the authors detect the user's interest in a webpage using two factors: the scrolling speed and the time spent. The weight of each factor value is calculated based on a predefined range (for example, when the scrolling speed is between x and y, the weight will be z). This transformation of the value into weight is not always accurate and excludes the diversity in user habits.

Makvana et al. [6] and Wu et al. [7] extract user interest from the user's query. The authors in [7] proposed an approach to solve the polysemy problem through query expansion. The approach collects keywords from the query, title, URL, content (snippets), and time spent of clicked websites (websites resulting from the query). Then, it computes the weight of each keyword using co-occurrence. Finally, the approach creates the user profile with the pairs (keyword, weight). The approach proposed by Hawalah et al. [9] represents the user interest in a model with a keyword and weight ($K_i$, $W_i$) pair vector. Each time the user enters a query, the approach extracts keywords and searches them in the profile. In the absence of a keyword, the approach adds it with a predefined weight ($W_i$); otherwise, the approach adds a unit score to $W_i$. The weights of all keywords decrease over time (current time (t) and last update time ($t_0$)) by the following formula :

$$W_{i\_new} = W_{i\_old} \times \lambda \text{ where } \lambda = e^{\log_2 \frac{t-t0}{30}} \qquad (1)$$

The two previous approaches [6][7] suffer from the same problems as the first approach [3], namely, they cannot distinguish between interesting and uninteresting keywords.

The architecture described in [9] creates the user profile through three phases. In the first phase, the architecture collects information such as visited websites, their content, time of the visit, and the duration. After the collection is done, the architecture fetches text from the webpages, removes all noise data from it (like HTML tags), tokenizes it, and removes stop words. Finally, each term is transformed into its stem. The resulting text is called a document. In the second phase, the architecture computes the TF*IDF weight (TF is the Term Frequency in a document, and IDF is the Inverse Document Frequency, which represents the number of documents containing the term divided by the total number of documents) of each term and creates a vector space that contains terms with weights. In the last phase, using cosine similarity, the architecture maps each visited website to the appropriate concept in the reference ontology. TABLE I summarizes the existing approaches.

The behavior of each user may differ from the others. Each user has their own reading speed (e.g., scrolling speed and the time spent). Therefore, the use of static intervals as in [5][8] is not practical since it does not take into account the diversity of users' behaviors. For instance, older users may spend more time than younger ones. This does not necessarily mean that they are more interested in this type of content, as it may occur due to reading difficulties. On the other hand, the existing solutions do not use factors [3][6][7] or enough factors [4][10] to determine the degree of user interest in a specific topic, which, in the meantime, affects the whole determination process and generates a false user interest.

To overcome this, we propose an approach that employs Fuzzy Logic. Instead of using a predefined range for all the users, each user's ranges will be calculated based on their browsing habits. We also introduce several factors to improve the detection process. Thus, this translates into high accuracy and adaptability. In this paper, we will consider the following aspects:

- We collect the browsing history with several parameters (factors) about each visited website, such as the time spent, the number of visits, and the scrolling speed.
- We apply the Fuzzy Logic in order to overcome the misinterpretation of factors weights and to provide better adaptability.

TABLE I. COMPARISON OF PROPOSED RESEARCHES.

| Authors | Method | Profile constructed based on | Factors | Information collection approach | Profilin method |
|---|---|---|---|---|---|
| Tchantchou and Ezin [3] | Multi-agent architecture | Browsing history | N.A | Hybrid | Content-based |
| Moawad et al. [4] | Multi-agent architecture | Browsing history User Behavior | User Actions | Hybrid | Content-based |
| Singh and Sharma [6] | Client/server Multi-agent architecture | Browsing history User Behavior | Scrolling speed Time spent | Implicit | Content-based |
| Makvana et al. [7] | Approach | User queries User Behavior | Time spent | Implicit | Content-based |
| Wu et al. [7] | Approach | User queries | N.A | Implicit | Collaborative-based |
| Hawalah and Fasli [9] | Approach | Browsing history User Behavior | Time spent | Implicit | Content-based |

## III. THE FUZZY INFERENCE SYSTEM

In this phase, we attempt to build a Fuzzy Logic system to predict the user interest degree and to solve the problem of misinterpretation of factors weights.

Fuzzy Logic was proposed first by Lotfi Zadeh in 1994 [10]. Unlike the binary logic, it does not use exact values to represent a situation (0 or 1, like or dislike, true or false). This type of logic represents the situation with a continuous value from 0 to 1, which gives the computer the ability to represent the unclear idea of humans, e.g., in the describing of a room brightness, instead of using a dark or a bright room (0 or 1), we can represent the degree of light and say little bright (0.6), little dark (0.4), very dark (0), very bright (1).

The Fuzzy Inference System (FIS) transforms multiple independent inputs into one output using Fuzzy Logic, memberships function, and rules. FIS has four components, the fuzzifier, the inference engine, the rule base, and the defuzzifier, as shown in Figure 1 [11]–[17].

### A. The Fuzzification

The fuzzification is the first phase in a Fuzzy Logic system that decomposes the crisp values into fuzzy sets. The fuzzification process has a few parameters to define. First, we define one or more imprecise fuzzy sets that divide the crisp values. Then, we represent the fuzzy sets using a membership function defined as follows:

$$(\mu_A: X \rightarrow \{0,1\} \mid X \in [values_{min}, values_{max}]) \qquad (2)$$

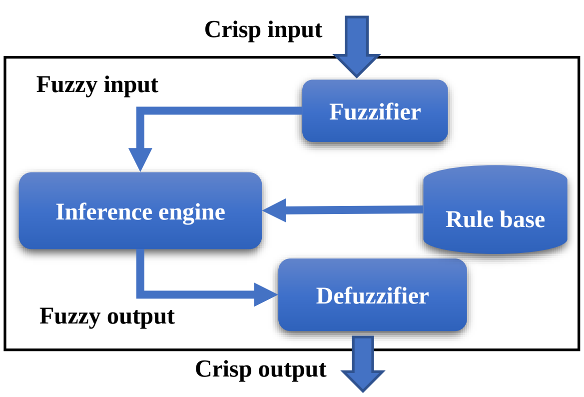


Figure 1. Fuzzy Logic system.

There are many membership functions, such as Triangular, Trapezoidal, Gaussian, and more. These functions assign the input value to one or more fuzzy sets with some degree of membership (Figure 2), e.g. if x = 40, the degree of membership of x is 0.3 in low and 0.3 in medium.

The above-mentioned misinterpretation of factor weight is generated from the predefined ranges. To resolve this problem using Fuzzy Logic, we calculate the range dynamically based on user browsing habits. The browsing values (e.g., scrolling speed) will be sorted by ascending order. Then, these values will be divided into three fuzzy sets that will be represented by the linguistic terms “low,” “medium,” and “high.” These sets will generate three intervals, where each of them will range from the minimum value of the set to the minimum value of the next one.

When users finish their browsing session, we extract the collected values (of those factors). Each value will be classified according to the previously generated intervals in order to determine the user interest degree in this type of content. These new values will be added to the previous ones and used to update the intervals, as shown in Figure 3. This allows the system to adapt to the user behavior and guarantee a high level of accuracy as compared to the existing solutions.

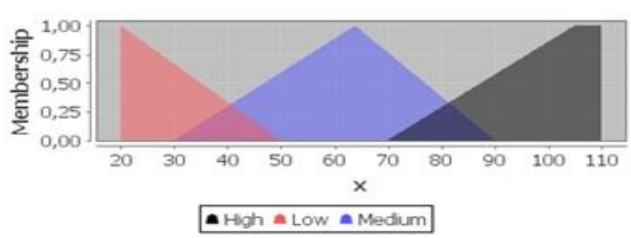


Figure 2. Triangular membership function example (Time spent in a Web site).

This adaptation transforms the captured value into a linguistic term to represent the right weight of this value (the linguistic term is more accurate than the value itself), which ensures the right detection of the user interesting topics. For example, let us consider two different users. Table A in Figure 4 shows the ranges of each user generated from the browsing habits. Now, let us assume that the two users will have the same browsing values for each factor (Table B in Figure 4). By using the fuzzification process on each factor value, we obtain different weights for each user according to the user's habits (Table C in Figure 4).

*B. The Inference Engine*

The Inference Engine is the core of the Fuzzy Logic system; this component is responsible for the calculation of one fuzzy output from a set of fuzzy inputs. The fuzzy output is calculated using a set of "IF…. THEN" rules built as follows:

**IF** *input1 is A* **AND** *input2 is B* **AND** *input3 is C* **THEN** *output is D*

The antecedent part of the rules contains the fuzzy inputs (input1 is A) obtained from the fuzzification process. A, B, and C represent one of the fuzzy sets of the first, second, and third variables, respectively (in our case, the variables are the factors such as scrolling speed, time spent, and the number of visits).

The consequence part of the rules contains the fuzzy output (output is D), which belongs to one of the following three fuzzy sets: uninteresting (range from 0 to 0.3), likely interesting (from 0.3 to 0.7), and interesting websites (from 0.7 to 1). The rules of the Fuzzy Inference Engine are presented in TABLE II ("I" represents Interesting, "LI" represents Likely interesting, and "UI" represents Uninteresting).

The Fuzzy Inference Engine maps the fuzzy inputs to the fuzzy output through two phases: first, it calculates the activation degree of each rule based on the fuzzified inputs. If the antecedent of the rule has more than one input, the engine applies the Fuzzy Logic operator (replace the and/or operator with the min/max between the two inputs) and composes those inputs. In the second phase, the engine aggregates the output of all rules into one fuzzy output. The aggregation is the union of all rule's outputs, which will be used in the next phase (the defuzzification).

*C. The Defuzzification*

The defuzzification is the inverse process of the fuzzification, which transforms the fuzzy output of the Fuzzy Inference Engine into a crisp value in order to make this result available to other applications.

The defuzzification is performed based on a decision-making algorithm that selects the best crisp value according to the fuzzy output. The two most used methods are the Center Of Gravity (COG), which return the center of the fuzzy output area and the Mean Of Maxima (MOM), which returns the crisp value or the mean of crisp values with the highest degree. In this paper, we used the COG function in the defuzzification process because the values generated by this function tend to change smoothly when there are small changes in the values of factors (the second produces two values that are far apart with slight changes in factors values).

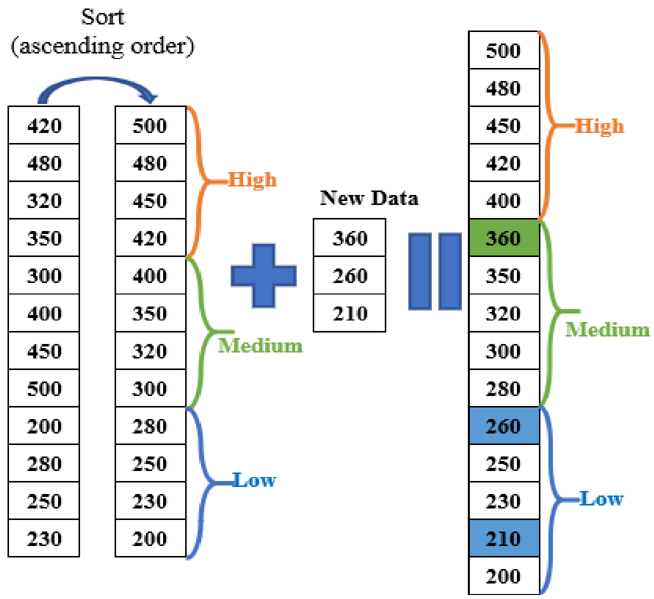


Figure 3. The adaptation process of intervals to user behavior.

## IV. USER PROFILE MODEL

The user profile is an essential component in this approach, which is why we must use a well-defined model to store it. This model describes the structure and the semantic relation between all information that exists in the profile.

There are several techniques to represent the user profile; we will discuss the more appropriate methods in our opinion based on the reviews of [17]–[20]. First, the Graphical models use modeling languages like Unified Modeling Language (UML) and Object-Role Modeling (ORM) to build the model. Then, it implements it using Structured Query Language (SQL), Non-Structured Query Language (NoSQL), or eXtensible Markup Language (XML) language. These models have a clear structure that makes it easy to retrieve information using queries in small data (queries become very complicated when the model contains a massive amount of data). Besides, these models do not support reasoning or context inference.

The Object-Oriented Models have the same principle as the Object-Oriented programming; they model the context and its relations with the others in a way similar to those (e.g., relations) between classes. The most important advantage of these models is the encapsulation (masks the context processing detail), and the reusability. However, it increases the number of needed resources and does not support reasoning.

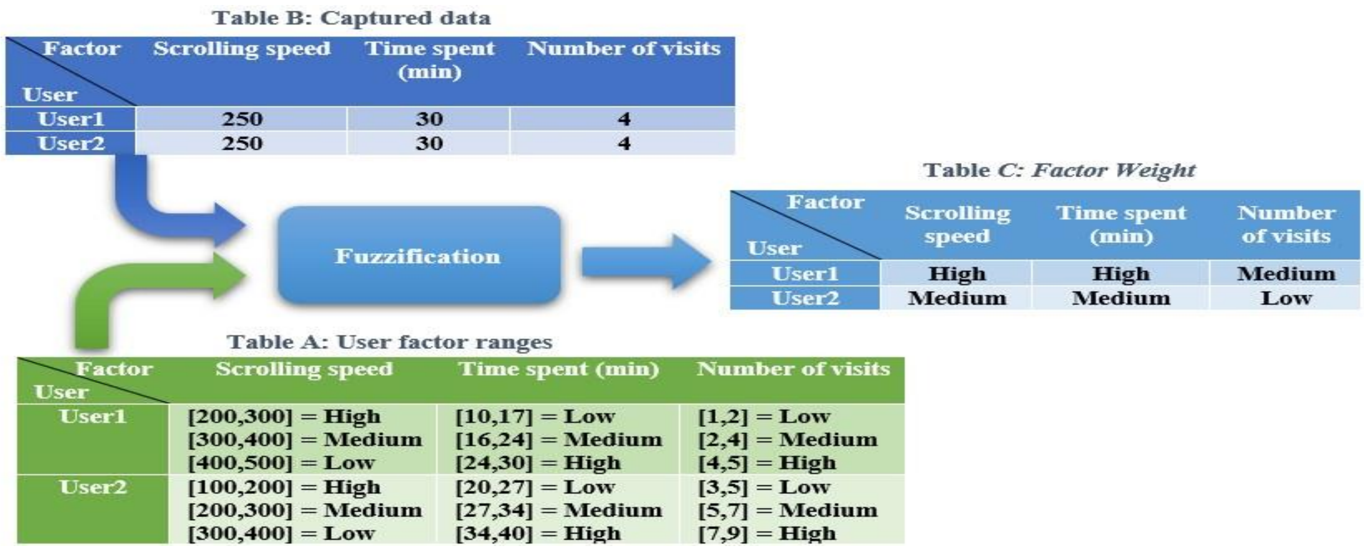


Table B: Captured data

| Factor / User | Scrolling speed | Time spent (min) | Number of visits |
|---|---|---|---|
| User1 | 250 | 30 | 4 |
| User2 | 250 | 30 | 4 |

Table C: Factor Weight

| Factor / User | Scrolling speed | Time spent (min) | Number of visits |
|---|---|---|---|
| User1 | High | High | Medium |
| User2 | Medium | Medium | Low |

Table A: User factor ranges

| Factor / User | Scrolling speed | Time spent (min) | Number of visits |
|---|---|---|---|
| User1 | [200,300] = High<br>[300,400] = Medium<br>[400,500] = Low | [10,17] = Low<br>[16,24] = Medium<br>[24,30] = High | [1,2] = Low<br>[2,4] = Medium<br>[4,5] = High |
| User2 | [100,200] = High<br>[200,300] = Medium<br>[300,400] = Low | [20,27] = Low<br>[27,34] = Medium<br>[34,40] = High | [3,5] = Low<br>[5,7] = Medium<br>[7,9] = High |

Figure 4. Fuzzification process.

The Logic-Based Models are based on binary logic. They use the facts, expressions, and rules to model the context (adds information as facts and removes/ modify it by rules). These models support reasoning and context inference. They have a very high degree of expressiveness and formality, and there are graphic tools for the development of this type of models. These models are heavily coupled with the application domain, which decreases their reusability.

The last one is the Ontology-Based models. These models represent the context with description logic such as Resource Description Framework (RDF), RDF Schema (RDFS), and Web Ontology Language (OWL). Those languages offer a high degree of expressiveness in the modeling of context and the modeling of relations between contexts. The ontology supports reasoning and inference (using inference engine like pellet), as well as separates the knowledge from the application, which increases the reuse and the share of knowledge between applications.

To model the user profile, we choose the ontology-based model for several reasons, such as the high expressiveness, many tools for implementation, the capability of reuse and share knowledge. Our ontology, represented in Figure 6, has two main classes:

- User Interests: contain user interest websites. This class has five attributes: URL of the website, scrolling speed, time spent, number of visits, interest degree calculated by our approach.
- Topic: represents the topic of the website. This class has only one attribute “Label” that represents the name of the topic (e.g., machine learning, sport).

The user profile model (classes, attributes, and the relation between classes) is created manually using Protégé [22] (a visual application to create an ontology) and maintained up to date automatically using the algorithm in Figure 5.

## V. CONCLUSION AND FUTURE WORK

The user profile contains information about the user that helps the customization systems to provide data or service to the user’s needs. In this paper, we propose an approach to automatically construct and update the user profile using a Fuzzy Logic system. This system solves the problem of factor weight misinterpretation and calculates the degree of interest of the user in specific topics. This paper contains the theory part of the system. This is a work in progress; the Fuzzy Logic system based on this approach is under development. As future works, we will develop the system, and perform the initial test with two users (we already have the data collected from those users) to prove the efficiency of this approach. Finally, we will discuss the possibility of increasing the number of factors.

```
Inputs:  New_Site, Interest_degree;
SS: Scrolling speed, TS: Time spent, NV: Number of visits
Begin:
 Profile = Get_User_Profile ();
 IF (Profile.Site_Exist (New_Site)) {
  Old_Site = Profile.Get_Site (New_Site.URL);
  Profile.Update (Interest_degree);
  Profile.Update (New_Site.SS, Old_Site.SS);
  Profile.Update(Average (New_Site.TS, Old_Site.TS));
  Profile.Update(Average (Old_Site.NV++));
  IF (Profile.Missing_Topics (New_Site.Topics)) {
   Profile.Update_Attribute (New_Site .Topics);
  }
 } Else {Profile.Add (New_Site, Interest_degree);}
End.
```

Figure 5. Algorithm to update the user profile.

TABLE II. FUZZY INFERENCE ENGINE RULES.

| Rule | IF | | | Then |
|---|---|---|---|---|
| | Scrolling speed | Time spent | Number of visits | Degree of interest |
| 1. | High | High | High | I |
| 2. | High | High | Medium | I |
| 3. | High | High | Low | I |
| 4. | High | Medium | High | I |
| 5. | High | Medium | Medium | LI |
| 6. | High | Medium | Low | LI |
| 7. | High | Low | High | I |
| 8. | High | Low | Medium | LI |
| 9. | High | Low | Low | UI |
| 10. | Medium | High | High | I |
| 11. | Medium | High | Medium | I |
| 12. | Medium | High | Low | LI |
| 13. | Medium | Medium | High | I |
| 14. | Medium | Medium | Medium | LI |
| 15. | Medium | Medium | Low | LI |
| 16. | Medium | Low | High | LI |
| 17. | Medium | Low | Medium | LI |
| 18. | Medium | Low | Low | UI |
| 19. | Low | High | High | I |
| 20. | Low | High | Medium | LI |
| 21. | Low | High | Low | UI |
| 22. | Low | Medium | High | LI |
| 23. | Low | Medium | Medium | UI |
| 24. | Low | Medium | Low | UI |
| 25. | Low | Low | High | UI |
| 26. | Low | Low | Medium | UI |
| 27. | Low | Low | Low | UI |

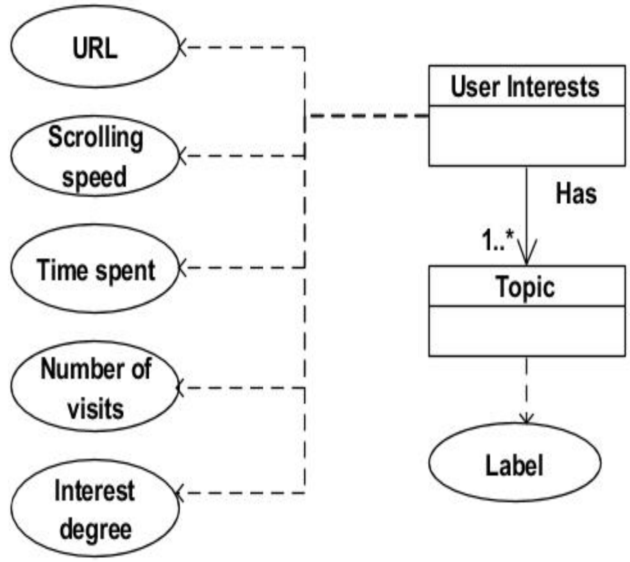


Figure 6. User profile model.

## REFERENCES


[1] S. Kanoje, S. Girase, and D. Mukhopadhyay, "User Profiling Trends, Techniques and Applications," vol. 1, no. 1, p. 6, 2015.

[2] A. Cufoglu, "User Profiling - A Short Review," *Int. J. Comput. Appl.*, vol. 108, no. 3, pp. 1–9, Dec. 2014, doi: 10.5120/18888-0179.

[3] Y.-U. S. Tchantchou and E. C. Ezin, "An Improving Mapping Process Based on a Clustering Algorithm for Modeling Hybrid and Dynamic Ontological User Profile," in *2017 13th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, Jaipur, India, 2017, pp. 1–8, doi: 10.1109/SITIS.2017.12.

[4] I. F. Moawad, H. Talha, E. Hosny, and M. Hashim, "Agent-based web search personalization approach using dynamic user profile," *Egypt. Inform. J.*, vol. 13, no. 3, pp. 191–198, Nov. 2012, doi: 10.1016/j.eij.2012.09.002.

[5] "WordNet | A Lexical Database for English." [Online]. Available: https://wordnet.princeton.edu/. [Accessed: 24-Feb-2020].

[6] A. Singh and A. Sharma, "A Multi-agent Framework for Context-Aware Dynamic User Profiling for Web Personalization," in *Software Engineering*, Springer, 2019, pp. 1–16.

[7] K. Makvana, P. Shah, and P. Shah, "A novel approach to personalize web search through user profiling and query reformulation," in *2014 International Conference on Data Mining and Intelligent Computing (ICDMIC)*, Delhi, India, 2014, pp. 1–10, doi: 10.1109/ICDMIC.2014.6954221.

[8] X. Wu, Y. Fu, S. Tian, Q. Zheng, and F. Tian, "A hybrid approach to personalized web search," in *Proceedings of the 2012 IEEE 16th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, 2012, pp. 214–220.

[9] A. Hawalah and M. Fasli, "Dynamic user profiles for web personalisation," *Expert Syst. Appl.*, vol. 42, no. 5, pp. 2547–2569, Apr. 2015, doi: 10.1016/j.eswa.2014.10.032.

[10] L. A. Zadeh, "Soft computing and fuzzy logic," in *Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems: Selected Papers by Lotfi a Zadeh*, World Scientific, 1996, pp. 796–804.

[11] Y. Bai and D. Wang, "Fundamentals of Fuzzy Logic Control — Fuzzy Sets, Fuzzy Rules and Defuzzifications," in *Advanced Fuzzy Logic Technologies in Industrial Applications*, Y. Bai, H. Zhuang, and D. Wang, Eds. London: Springer London, 2006, pp. 17–36.

[12] H. R. Berenji, "Fuzzy logic controllers," in *An introduction to fuzzy logic applications in intelligent systems*, Springer, 1992, pp. 69–96.

[13] D. Veit, "Fuzzy logic and its application to textile technology," in *Simulation in Textile Technology*, Elsevier, 2012, pp. 112–141.

[14] R. S. Jaiswal and M. V. Sarode, "An Overview on Fuzzy Logic & Fuzzy Elements," *Int. Res. J. Comput. Sci.*, vol. 3, no. 2, p. 6, 2015.

[15] A. K. Nandi, "GA-Fuzzy Approaches: Application to Modeling of Manufacturing Process," in *Statistical and Computational Techniques in Manufacturing*, J. P. Davim, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 145–185.

[16] S. N. Mandal, J. P. Choudhury, and S. R. B. Chaudhuri, "In Search of Suitable Fuzzy Membership Function in Prediction of Time Series Data," vol. 9, no. 3, p. 10, 2012.

[17] P. Cingolani and J. Alcalá-Fdez, "jFuzzyLogic: a Java Library to Design Fuzzy Logic Controllers According to the Standard for Fuzzy Control Programming," *Int. J. Comput. Intell. Syst.*,


 

vol. 6, no. sup1, pp. 61–75, Jun. 2013, doi: 10.1080/18756891.2013.818190.

[18] T. Strang and C. Linnhoff-Popien, “A context modeling survey,” in *Workshop Proceedings*, 2004.

[19] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, “Context Aware Computing for The Internet of Things: A Survey,” *IEEE Commun. Surv. Tutor.*, vol. 16, no. 1, pp. 414–454, 2014, doi: 10.1109/SURV.2013.042313.00197.

[20] X. Li, M. Eckert, J.-F. Martinez, and G. Rubio, “Context Aware Middleware Architectures: Survey and Challenges,” *Sensors*, vol. 15, no. 8, pp. 20570–20607, Aug. 2015, doi: 10.3390/s150820570.

[21] C. Bettini *et al.*, “A survey of context modelling and reasoning techniques,” *Pervasive Mob. Comput.*, vol. 6, no. 2, pp. 161–180, Apr. 2010, doi: 10.1016/j.pmcj.2009.06.002.

[22] “protégé.” [Online]. Available: https://protege.stanford.edu/. [Accessed: 12-Mar-2020].