Dynamic User Profiles for Web Personalization

Ahmad Hawalah[a,b], Maria Fasli[a,*]

[a]*School of Computer Science and Electronic Engineering, University of Essex*
Wivenhoe Park, Colchester CO4 3SQ, UK
[b]*College of Computer Science and Engineering, University of Taibah, Medina, Saudi Arabia*

Abstract

Web personalization systems are used to enhance the user experience by providing tailor-made services based on the user's interests and preferences which are typically stored in user profiles. For such systems to remain effective, the profiles need to be able to adapt and reflect the users' changing behaviour. In this paper, we introduce a set of methods designed to capture and track user interests and maintain dynamic user profiles within a personalization system. User interests are represented as ontological concepts which are constructed by mapping web pages visited by a user to a reference ontology and are subsequently used to learn short-term and long-term interests. A multi-agent system facilitates and coordinates the capture, storage, management and adaptation of user interests. We propose a search system that utilizes our dynamic user profile to provide a personalized search experience. We present a series of experiments that show how our system can effectively model a dynamic user profile and is capable of learning and adapting to different user browsing behaviours.

Keywords:
dynamic user profile, modelling user behaviour, web personalization, ontology, multi-agent systems

1. Introduction

The dramatic growth of information on the WWW has inadvertently led to information overload and hence finding a specific piece of information has become difficult and time consuming (Challam et al., 2007). Web personalization systems have emerged in recent years in order to deal with this problem aiming to provide a personalized experience to users based on their individual preferences, interests and needs. Such systems have been developed for different domains of application (Pignotti et al., 2004; Challam et al., 2007; Sieg et al., 2007; Pan et al., 2007). In e-commerce, such systems have been used to recommend new items and products to users based on their previous purchasing history (Gorgoglione et al., 2006; Huang et al., 2004), while in e-learning, they are used to provide personalized e-learning services (Sun and Xie, 2009; Zhuhadar and Nasraoui, 2008). Personalization systems require and maintain information about users, and this may include demographic data, interests, preferences, and previous history. One of the main challenges in such systems is that user interests, preferences and needs are not fixed, but change over time. If user profiles contain just static information, this eventually leads to constraining the personalization process and recommending irrelevant services and items over time. To overcome this problem, methods are required for learning and understanding different user behaviours, and then adapting the profiles accordingly.

[*]Corresponding author
Email addresses: `ahawalah@taibahu.edu.sa` (Ahmad Hawalah), `mfasli@essex.ac.uk` (Maria Fasli)

In this paper, we utilize ontological profiles to capture user interests and provide recommendations based on these. The contribution of our work is threefold. Firstly, we introduce two algorithms in order to improve the mapping process between web pages visited by the user that contain implicit information about the user interests and a reference ontology to explicitly represent these interests. Secondly, we introduce novel techniques to construct ontological short-term and long-term profiles that are tailored to the users, and adapt them based on their ongoing behaviour. Thirdly, the methods introduced attempt to recognize and handle potential interest drift and interest shift in the user interests.

To demonstrate our work, we introduce a personalization system that consists of three phases. The first phase is the information retrieval phase which involves preparing a reference ontology, collecting user navigation behaviour, and mapping visited web pages to the reference ontology. Indeed, this phase is very important as capturing inaccurate user interests would directly affect the subsequent phases and eventually the personalization performance. In this phase, we utilize two novel algorithms based on our work in (Hawalah and Fasli, 2011a) to improve the mapping process. The second phase is the profile adaptation and learning phase which utilizes previous work in (Hawalah and Fasli, 2011b). This phase plays a major role in our model as it is responsible for learning, adapting and modelling ontological-user profiles. This also includes methods to adapt the ontological profiles to any shift or drift that might occur in the users' behaviour. This phase makes use of a multi-agent system that coordinates the various processes and ensures that the user profile remains up-to-date. In the last phase, a re-ranking search system is introduced that utilizes the dynamic user profile to provide a personalized search experience. The re-ranking search system takes advantage of the user interests to provide more personalized search results. To evaluate this work, we have conducted experiments with users over a period of time to assess the ability and effectiveness of our methods in tracking and adapting to changes in the user behaviour.

The rest of the paper is structured as follows. First we discuss related work. Section 3 presents the main architecture for modelling dynamic user profiles which consists of three phases with each phase being discussed in more detail in a subsequent section. We introduce the evaluation in section 4. Section 5 describes the evaluation of the mapping and profile construction methods, while section 6, details the evaluation of the dynamic user profiling methods in the context of their deployment within a personalized search system. The paper ends with the conclusions and pointers to future work.

2. Related Work

As information on the WWW continues to proliferate, users find it increasingly difficult and time-consuming to sift through this information. To aid the user in his/her quest for the right information (be it on items, products, movies or articles), recommender and web personalization systems have emerged. Recommender systems use two broad categories of techniques: content-based and collaborative-filtering (CF) techniques. The first technique views users as individuals. Such systems track the user interests and preferences and create an explicit profile that characterizes the user and any ensuing recommendations are guided by the profile. The second technique does not make use of complex user profiles, instead information in the form of ratings (for instance on a scale from 1-5) for items, products, etc. is collated from each user and then similarities between the users and items are calculated and exploited to make new recommendations. Our work in this paper, is related to the first family of techniques used for personalization systems.

Personalization systems may rely on different knowledge bases to learn and model user profiles including taxonomies (Eirinaki et al., 2006; Mooney et al., 1998), flat databases (Wu et al., 2001) and ontologies (Middleton et al., 2004; Weng and Chang, 2008; Felden and Linden, 2007; Liu et al., 2008). Different techniques have been proposed to discover and recommend new items to users based on the modelled profiles, such as using content-based models (Liu et al., 2008; Mooney and Roy, 2000; Middleton et al., 2004), spreading activation techniques (Blanco-Fernandez et al., 2011; Liang et al., 2008; Gao et al., 2008; Weng and Chang, 2008) and classification techniques (Xu et al., 2008).

Ontologies encapsulate knowledge about a domain of application (Razmerita and Lytras, 2008) and as such they provide a highly expressive medium for describing user interests and preferences and rich interrelations among them. Unlike simple methods of representing information such as weighted keywords and semantic network profiles, an ontology provides a more powerful, deeper and broader concept hierarchy representation for user profiles (Gauch et al., 2007). Using ontologies to model profiles has already been proposed in various applications in the field of information systems in general and personalization in particular. Trajkova and Gauch (2004) and Zhang et al. (2007) for example, introduced a general mechanism for modelling user profiles by implicitly tracking user visited web pages. These visited web pages are then mapped to different concepts in the Open Directory Project (ODP) domain ontology[1].

In (Weng and Chang, 2008), user browsing and search activities are tracked and processed in order to build user profiles. The user profiles are learnt based on an ontology that consists of a hierarchical representation of different topics. A spreading activation model is then applied using this ontology to provide adequate recommendations to users. Middleton et al. (2004) described two hybrid recommender systems that employ ontological user profiles to recommend appropriate research papers to academic staff and students. The novelty in this work is in the ability of these profiles to infer more preferences based on the collected user data, while it also offers users a visualization of their profiles so that they can explicitly modify them. Sieg et al. (2007) presented an ontological user profile for tracking user behaviour in a web search environment. A spreading activation mechanism was proposed to learn and maintain user interests. User profiles are then used to re-rank the search results based on the users' current interests. In (Challam et al., 2007), a web search system based on ontological user profiles is proposed. The authors suggest that using contextual information, i.e. the user's current task, can provide a more effective personalized experience. However, these studies do not focus on the process of learning user behaviour over time; instead they focus on providing accurate search personalization at a particular time. Anand et al. (2007) proposed using an ontology to represent user interests. In this approach, the content descriptor of the items in an ontology is used with user actual ratings to provide a recommendation.

Some studies have proposed sophisticated approaches for modelling user profiles in a more dynamic way. Grcar et al. (2005) presented a dynamic user profile that tracks implicitly user navigation and consists of short and long-term models. However, this research assumed that all the visited web pages are of interest to the user no matter how long s/he spends reading through them. As all pages are treated the same, the potential strength of interest in various topics is ignored. Another issue with this study is that the short and long-term folders have been limited to predefined sizes (i.e. 5 and 300 respectively). The short-term folder stores the most recent visited web pages, while the long-term folder stores the last viewed web pages. This technique limits the user profile to a small number of interests, and at the same time can suffer from instability due to highly changeable interests. Along the same lines, Li et al. (2007) proposed a short-term model that uses a page-history buffer (PHB) which emulates the functionality of the cache and database disk buffer. Again the size of the buffer is fixed, which leads to the same problems as in (Grcar et al., 2005). Another limitation relates to the representation of the user interests in the profiles as the interest-topic and associated weight. An interest's weight is represented as the total number of visits for this interest. The model does not make use of a time discount factor, hence, the weight of the interest would remain the same over time. For example, if a user visited pages associated with the topic *IPhone* 20 times over a period of time, then this number remains the same even if the user has shown no interest in *IPhone* in the more recent past.

[1]http://www.dmoz.org/

Cantador et al. (2008) proposed an adaptation strategy for a dynamic user profile. This work deploys a reference ontology that is used to map user preferences and current context to provide personalization. User preferences can be seen as users' long-term interests, while current user context represents the users' interests at the current runtime. All user interests are represented as weighted concepts. However, user preferences are stored in a stack history which is limited to a predefined size and as such it would only be able to store a small proportion of user history but not all user preferences. Furthermore, the mechanism for detecting current user context does not consider the user's previous sessions. For instance, if a user shows interest in a concept *Car* in many sessions, and shows interest in a concept *Football* in the last session, both of these concepts would be treated similarly without taking into account that a user might be more interested in *Car* as s/he has shown an interest in this concept over many sessions.

Although the above studies provide different ways to model user profiles, most of them do not deal with the user's changing interests over time, while others only attempted to model low-level dynamic user profiles. The distinction made in these works between short and long-term interests, and interests at current runtime is often blurred and most such approaches treat users as having fixed size interests.

3. Capturing and Modelling Dynamic User Profiles

In this section, and following on from the identification of a number of drawbacks in existing works in personalization, we propose a dynamic user profiling approach. In developing our research, we have taken into account a number of factors and desirable properties for personalization systems:

- Users are reluctant to provide information about their interests and preferences explicitly through completing questionnaires, etc. (Montaner, 2001). Hence, a flexible personalization system should try and capture interests through tracking user behaviours implicitly without user intervention, but as accurately as possible.

- In order to be able to provide advanced services to users, their interests and preferences should be captured and stored in an appropriate format that would enable further processing to be performed and useful information to be extracted. The use of ontologies has been shown to provide a significant improvement in the performance of personalization systems (Challam et al., 2007; Trajkova and Gauch, 2004; Middleton et al., 2004). To this end, we have decided to deploy ontologies for representing user interests to support more effective, and semantically-driven exploration of the profile and hence richer recommendations.

- User interests very rarely remain static, they constantly change and evolve over time. Most existing personalization systems do not deal with this challenge in capturing user behaviour or they do not do so in a way that looks at the user behaviour in a holistic way. Our aim is to develop profile methods that should be able to track the change in the user interests including any interest-shift or drift and adapt accordingly.

- As discussed in the previous section, user interests can be distinguished into short-term, which are the user current interests and can be highly changeable, and long-term, which tend to be more stable over time. Such a distinction of user interests is useful to capture and it provides an extra dimension in trying to understand the individual user and his/her needs. However, this distinction is very often blurred in existing approaches or not captured at all. We aim to capture this aspect of user interests in our system in a clear manner.

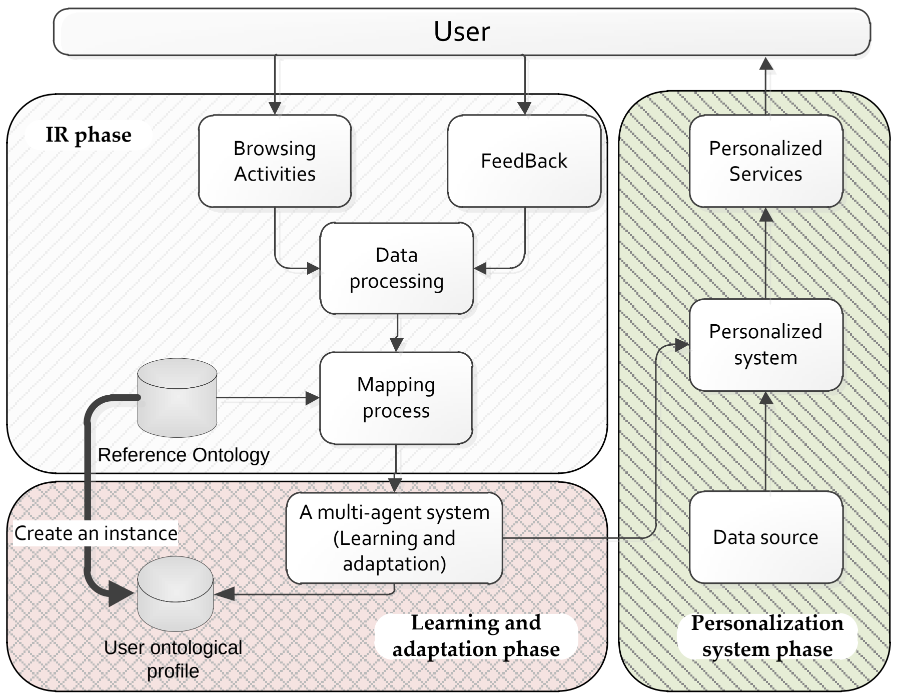


Figure 1: The phases and architecture of the personalization system.

- Typically work in personalization is developed to address specific domain problems. We aim to develop models and methods that will have wider applicability and can be integrated and deployed to provide a range of personalization services.

Building upon these, we propose a novel personalization system that consists of three phases as illustrated in Figure 1:

- The Information Retrieval phase.
- The Learning and Adaptation phase.
- The Personalization phase.

3.1. Information Retrieval (IR) Phase

The aim of this phase is to collect user browsing behaviour to discover their interests. Personalization is aimed at enhancing the process of retrieving information as a user receives tailored contents to his/her interests, but if the system collects the wrong interests, then it would provide inaccurate and ineffective services. The tracking and discovery of user interests is done in three stages.

3.1.1. Stage One: Tracking User Browsing Behaviour

Since collecting data explicitly adds more of a burden on users (Montaner, 2001), we aim at collecting such behaviour unobtrusively. The data that we need to observe in this system is the visited websites, contents of each web page, timestamp which denotes the date/time of the visit and finally the duration of the visit. The *Browsing Activities* component in Figure 1 is used to collect and then store this information in a log file. This component records all the visited web pages $W = \{W_1, W_2, ..., W_n\}$ during the user browsing sessions and fetches all the textual contents tx_i for each $W_i \in W$. For each visited web page W_i, the timestamp q is recorded as well as the duration k of the user's visit to the web page. This raw information is stored in a *Raw* log file which is then used by the *Data-processing* component. First, all the noise in tx_i such as HTML tags is removed[2]. This component then tokenizes all the tx_i for each W_i to discover terms t_i. Once all the tx_i are cleaned and tokenized, it is important to reduce the dimensionality of terms by removing all unnecessary ones which have low discriminating values as well as reducing their ambiguity. Hence, we first remove all the common terms such as 'and', 'or' and 'the' (stop words) using a stop list. Subsequently, we apply the Porter stemming algorithm (Porter, 1997) on all the t_i in order to return each term to its stem (e.g. *computer* to *compute*). The outcome of this phase is a *P-log* file that can be processed further.

3.1.2. Stage Two: Reference Ontology Preparation

Ontologies comprise rich knowledge representation structures and their use has been shown to provide a significant improvement in the performance of personalization systems (Challam et al., 2007; Trajkova and Gauch, 2004; Middleton et al., 2004). In the proposed system, an ontology plays a significant role in modelling dynamic user profiles. A reference (or domain) ontology describes a particular domain of application and is usually modelled in a hierarchical way in which super-classes are linked to sub-classes. Each concept in an ontology is typically associated with a document that explains and represents it. Unlike flat representations, a reference ontology provides a richer representation of information in that semantic and structural relationships are defined explicitly. We use a reference ontology for two purposes: firstly, to identify user interests based on the visited web pages, and secondly, to represent user interests as ontological concepts.

Let O be a reference ontology that represents a domain and C a set of concepts that belongs to O. $R = \{r_1, ...r_n\}$ is a set of relations that links two concepts. A reference ontology might have different types of relations such as r_1: 'is-a', r_2: 'has-a', etc. As we propose a content-based personalization system, each $c_i \in C$ in O is associated with a document d_i that represents the c_i.

We first convert all the terms in all the documents associated with concepts in the reference ontology to vectors using the $TF * IDF$ classifier. That is, each term $t_j \in d_i$ where $d_i \to c_i$ (associated to c_i) in O is given a weight value w that measures the importance of the term t_j within a document d_i. Using the $TF * IDF$, this weight can be computed by dividing the number of occurrences of term t_j by the total number of terms $\mid t \mid$ in a document d_i. Then the inverse document frequency (IDF) which represents the general importance of the term is calculated by dividing the total number of documents D in O ($\mid D \mid$) by the total number of documents d_i that contain the term t_j in O $\mid t_j \triangleright d_i \in D \mid$. Finally, each term in the ontology is given a weight from 0 to 1.

$$\begin{aligned} w_{t_j} &= (tf_{ji} * idf_j) \\ idf_j &= \log(\frac{\mid D \mid}{\mid t_j \triangleright d_i \in D \mid}) \end{aligned} \tag{1}$$

Although the vector space model is one of the simplest classification methods, it has one drawback: it distinguishes between terms or vectors that have the same root. Words such as 'play', 'plays' and 'played'

[2]Although works have shown that HTML structure can be used as part of the indexation process, in this phase of our work, we do not make use of such information.

are processed as different words. This makes the classifier less effective as the dimensionality of the terms increases. In order to avoid this, and similarly to the first stage, we remove stop words and apply the Porter stemming algorithm (Porter, 1997) to remove term suffixes and return each term in the ontology to its stem. Once all the terms in a reference ontology are converted to vectors, the ontology will be ready to be used to discover user interests from user visited web pages.

3.1.3. Stage Three: Mapping Documents to a Reference Ontology

Once the term weights are calculated for each term in the ontology, any vector similarity method can be used to map visited web pages to appropriate concepts (or classes) in the reference ontology. In this paper, the well-known cosine similarity algorithm (Baeza-Yates and Ribeiro-Neto, 2011) is applied to classify web pages to the right concepts. Cosine similarity is used to ascertain the similarity between two vectors and measures the cosine of the angle between them – in our case, the vector representing the web page visited against the vector representing a specific concept in the reference ontology.

$$sim_{Cosine}(d_1, d_2) = \frac{\sum_{i=1}^{n} w_{i1}, w_{i2}}{\sqrt{\sum_{i=1}^{n} {w_{i1}}^2} * \sqrt{\sum_{i=1}^{n} {w_{i2}}^2}} \tag{2}$$

Several studies have tried to improve the accuracy of the mapping process. One such approach is by using an ontology with a limited number of levels. Liu et al. (2002) for instance, map user interests to a reference ontology that is limited to just two levels. Studies such as (Speretta and Gauch, 2005; Chen et al., 2002; Trajkova and Gauch, 2004) use a three level ontology to map user interests. When it comes to the retrieval precision, using a limited number of levels has been reported to improve the overall accuracy as it reduces the number of concepts in a reference ontology which in turn minimizes the error of mapping contents to wrong concepts. However, levels two or three of an ontology can be too general and broad to represent actual user interests, and hence this risks no specific interests being recognized. For instance, in the Open Directory Project (ODP)[3] ontology, level two *Computers/Programming* or three *Computers/Programming/Languages* are too general to represent interests such as *Java* or *C#* programming languages.

Another method to improve the mapping performance is by adding a pre-defined percentage of each sub-class's weight to its super-class. The idea is that if a user is interested in a particular class, then s/he is also interested in its super-class. For example, if a user is interested in football, then s/he also has some interest in its super-class which is likely to be *Sport*. Middleton et al. (2004) and Kim et al. (2007) implemented this idea by adding an extra 50% for each interest to its super-class. The process is then repeated until the root class. Although this method showed an improvement over the original cosine similarity, the accumulation behaviour that it induces puts more emphasis on the top level classes which are too general to represent actual user interests, while middle and low level classes receive less attention.

Daoud et al. (2008) concur that representing interests with level two of an ontology is too general, while a leaf node representation is too detailed. They suggested that the most relevant concept is the one that has the greater number of dependencies. To this end, they proposed a sub-concept aggregation scheme where the main goal is to represent all user interests with level three in an ontology. The weight of the level three in this system is calculated by adding the weights of all its sub-concepts. However, representing user interests by level three in an ontology is restrictive as ontologies can have different structure and complexity. Some ontologies may have few levels, while others may extend to several; the ODP has seven levels.

To address the limitations of the aforementioned methods, we need a method that improves the mapping process and at the same time maintains both general and specific interests. To this end, we introduce two

[3]http://www.dmoz.org/

new methods called Gradual Extra Weight (GEW) and Contextual Concept Clustering (3C) (Hawalah and Fasli, 2011a).

The Gradual Extra Weight Algorithm (GEW). The GEW algorithm is based on the idea that if a user is interested in a particular class, then s/he also has some interest in its super-class. However, we make no assumption about the number of levels an ontology has. Moreover, we do not assign a specific percentage of a sub-class to be added to its super-class such as in (Middleton et al., 2004) and (Kim et al., 2007).

We propose an auto-tuning mechanism in which the percentage value of each sub-concept that is added to its super-concept is tailored to different levels of the ontology. In this mechanism, we assume that the concepts deeper in the ontology express more specific interests than a general one which is expressed with a concept higher up in the ontology (e.g. *Java* as opposed to *Programming Languages*). Therefore, the concepts in the lower levels would add more weight to their super-concepts than those in higher levels. Equation 3 shows how the extra percentage (EP) for each level is calculated in order to transfer weight:

$$EP = \frac{c_i.Current\_level * \alpha}{O.Max\_levels} \tag{3}$$

Where $c_i.Current\_level$ is the level of a sub-concept, α is a parameter that controls how much weight is transferred from one concept to another (i.e. $\alpha = 0$ means no weight is transferred, while $\alpha = 1$ means the maximum weight is transferred) and $O.Max\_levels$ is the total number of levels in the ontology O.

Using the total number of levels in an ontology in equation 3 allows us to gradually reduce the percentage of weight that is transferred as we move up towards the root and keep a balance between general and specific interests. The default α value is 0.5, as in this case the maximum weight transferred from the concepts is 50% from the leaf concept to its super-concept and this percentage is reduced as we move up towards the root. However, α is not fixed but it can be changed based on the ontology and the number of levels at hand. When changing the α value, extra care is required to avoid selecting too high or too low an α value, as the former might cause inflation at top levels, while the latter would make the GEW ineffective as the weight which is transferred from a concept to its super-concept would be too small and insignificant.

Finally, the GEW algorithm is applied to each visited web page after the cosine similarity is computed between this web page and all concepts in an ontology. However, one problem that might arise at this stage is that the ontology might have a huge number of concepts and hence, it would be too expensive to apply the GEW to all concepts. To reduce the computational load, we only apply the GEW on just the top γ results that have the highest similarity weights. Determining the best value for γ allows us to remove the concepts that do not add any value when applying the GEW algorithm[4]. The GEW algorithm is explained in more detail in Figure 2.

To demonstrate how the GEW algorithm works, consider a reference ontology as in Figure 3 with 6 levels. For each visited web page that is mapped to a reference ontology using the cosine similarity measure, all the concepts in this ontology would be assigned a similarity weight $\in [0, 1]$. The GEW algorithm starts from the highest similar concept which in this example is the concept with ID 6 that has a weight of 0.3. The extra weight based on the level of this concept which is level 6 is computed as in the dashed-box on the left. The EP which is 0.5 is then multiplied by the original weight of this concept which is 0.3. The total extra weight (0.15) will be added to the upper-concept. As the similarity of this upper-concept and the visited web page is 0.12, this weight will be modified by adding the extra 0.15 to it, so the new weight is 0.27. This process is then repeated until the root node is reached.

[4]In the evaluation section, we describe how the value for γ can be identified experimentally.

```
Algorithm 1: GEW algorithm
$CON = \{c_1, ..., c_n\} \in O$ //Concepts in an ontology $O$.
$c_i\_Level$ = is the level of a concept $c_i$.
$c_i\_d$ = is the document that represents the $c_i$ concept.
P-log=$\{w_1\_ts, w_2\_ts, ..., w_n\_ts\}$ // P-log holds web pages and their textual contents.
$EP$= extra percentage
$Can_{List}$ = a list which holds all the candidate concepts that might be mapped to a web page.
Input: All $w \in$ P-log and a reference ontology $O$
Output: Candidate concepts $CON$ for each $w_i \in$P-log
begin
    // (1) Applying the cosine similarity measure between all $w \in$P-log and $CON \in O$;
    foreach $w_i\_ts \in P\text{-}log$ do
        Calculate TF*IDF;
        foreach $c_i \in O$ do
            Compute $sim_{Cosine}(w_i\_ts,\ c_i\_d)$ ;
            $Can_{List}\_Add(w_i\_ts,\ c_i\_d,\ sim)$ // add the web page, concept and weight to the list;
        // (2) Applying the GEW on the top $\gamma$ concepts;
        $Can_{List}\_Sort$ by the cosine similarity weight in a desc order;
        count=1;
        foreach $c_i \in Can_{List}$ and $count \leq \gamma$ do
            $cc = c_i$ // it is a variable that stores the current class that is being processed;
            while $cc\_sim\_weight > 0$ and $cc \neq Root$ do
                Compute $EP = \frac{cc\_Level * \alpha}{O\_Max\_levels}$;
                Extra_weight = $EP * cc\_sim\_weight$ ;
                $Can_{List}$_Find($cc$_Super-class);
                $cc$_Super-class+= Extra_weight;
                $cc$ = $cc$_Super-class;
            count++;
        $Can_{List}\_Sort$ by the cosine similarity weight in a desc order;
```

Figure 2: The GEW algorithm.

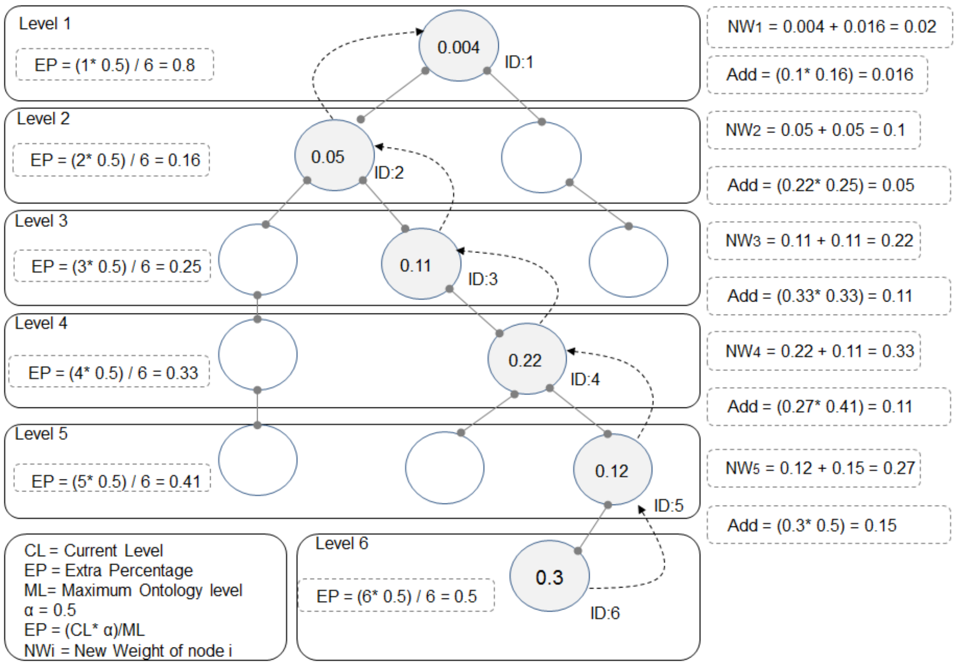


Figure 3: An example of the GEW algorithm.

The Contextual Concept Clustering Algorithm (3C). Although the GEW method may improve the process of mapping web pages to concepts, correct mapping cannot be guaranteed as not all the visited web pages usually have good representative contents or web pages may contain a lot of noise.

We aim to further improve the mapping process by taking advantage of having a log file that stores the user's browsing behaviour. When users browse, they tend to visit several web pages that represent one interest during the same session. All current works in the field of personalization that map each visited web page to a reference ontology, do so in isolation and without any consideration about other web pages that have been visited during the same session. However, a set of visited web pages in one session might hold important hidden information that can be exploited to improve the mapping process. Assume that a user has visited 3 web pages about the XML topic, and some of these pages contain contents that are not very representative. A traditional mapping approach would map each web page to a concept from an ontology separately. As some of these web pages have poor representative contents, such an approach might assign wrong concepts to them. But if we grouped similar web pages together and applied the mapping process on this group, the risk of mapping these web pages to the wrong concept could potentially be reduced. This is because the web pages with good representative contents can "hold up" the pages with poor contents and eventually help the mapping process to map all the pages in this group (and even the ones with noise) to the correct concept in the reference ontology.

Another reason why grouping web pages may be effective is that such a grouping may in essence better encapsulate the *context* of the user behaviour. Consider a user that visits a web page that provides a comparison between the C# and Java languages and hence contains two concepts. An assignment to either of these would be correct from the mapping process point of view. However, if this user is mainly interested in the C# language and is just curious to find out why it might be better than Java, s/he would not be interested in receiving recommendations about Java. Although classifying such a web page to either

C# or Java would be conceptually correct, it would not be appropriate for this specific user who is mainly interested in C#. Therefore, taking into account the user session (group of pages) would help to classify them to the most appropriate concept; in our example the C# concept.

To address these issues, we introduce the Contextual Concept Clustering (3C) algorithm which aims to group related web pages into clusters which are then assigned to a particular concept in a reference ontology. The 3C algorithm works as follows (Figure 4): 1) For each browsing session, the GEW algorithm is applied in order to find the top γ similarities between each visited web page and concepts in the reference ontology. 2) After applying the GEW, the highest β similar concepts from the ontology to each web page are used as possible candidates to represent it. We need to consider all the top β results because in some cases the concept with the highest similarity value does not give a correct representation of a web page. This could be due to poor or irrelevant information in a web page, or it could be simply due to a high level of noise. 3) The "context" is then exploited by selecting the common concept that is associated with different web pages. This common concept eventually is selected to represent a web page. If there is no common concept, the concept with the highest similarity weight is selected. The β value can be identified by conducting an experiment to find the most appropriate value (see evaluation section).

To illustrate the 3C algorithm, consider a user who has visited six web pages as in Figure 5. First the GEW algorithm is applied to all these web pages and then the top β similar concepts are selected as possible candidates to represent this concept. In this example, each web page is linked to 5 concepts (concepts are represented as circles and unique id numbers from the ontology). The 3C mechanism clusters these top similar concepts into groups where each group contains a set of web pages that have the same common candidate concept mapped to them. Hence there are three groups. Group 1 consists of four web pages that share concept 25 as common. The weight of this group can then be computed as the total of all the semantic similarity weights between the common concept 25, and all other web pages that are members of group 1 namely web pages 1, 2, 3 and 4. However, web pages can also be mapped at the same time to other candidate concepts that might be clustered in different groups. For example, web pages 1 and 2 can be assigned to either group 1 or 2, while web page 3 can be assigned to either group 1 or 3. If there are more than one groups, the page is assigned to the group with the highest total weight. So, in this example web pages 1, 2, 3 and 4 would be represented by concept 25, while web pages 5 and 6 would be represented by concept 21.

3.2. Learning and Adaptation Phase

Typically, user interests change over time as they may lose/acquire interest in an object. If a user profile is not able to detect and adapt to such changes, this will inevitably affect the personalization performance. According to (Cantador et al., 2008; Picault and Ribiere, 2008), a number of heuristics can be used when modelling a user profile to recognize patterns in browsing behaviour:

- A concept may occur in a short period with a very low occurrence. This should not be considered to be a concept of interest to a user. For example, a user might open a web page that does not meet his needs so he closes it immediately.

- A concept may occur in a short period and its occurrence is very high. This concept should be considered as a short-term interest, and it should be removed once the user shows no more interest in it. For example, a user may be planning a day out in London which would involve looking for train tickets, museum and theater tickets. The *London* interest will exist for a while, but once the trip is taken, the user would have no further interest in London.

```
Algorithm 2: 3C algorithm
$CON = \{c_1, ..., c_n\} \in O$ //Concepts in an ontology $O$.
P-log=$\{w_1\_ts, w_2\_ts, ..., w_n\_ts\}$ // P-log holds web pages and their textual contents.
$Cl_{List} = \{c_1, ..., c_n\}$ a cluster list that contains the common concepts linked to web pages and
the total similarity weights.
$Can_{List}$ = a list which holds all the candidate concepts that might be mapped to a web page
and their cosine similarity weights.
$F_{List}$ = the final list that contains the final results of applying the 3C approach.
Input: $Can_{List}$ that holds all the candidate concepts after applying the GEW
Output: Web pages mapped to concepts from a reference ontology
begin
    // (1) Selecting top $\beta$ concepts with highest weights for each web page from the $Can_{List}$;
    foreach $w_i \in Can_{List}$ do
        Select top $\beta$ concepts, and remove the rest from the $Can_{List}$;
    // (2) Finding all concepts that appear in the $Can_{List}$;
    foreach Distinct $c_i \in Can_{List}$ do
        Cluster_total_weight= 0;
        $Cl_{List}$_Add($c_i$, Cluster_total_weight);
    // (3) Processing each cluster in the $Cl_{List}$;
    foreach $c_i \in Cl_{List}$ do
        foreach $w_i \in Can_{List}$ do
            if $w_i$_mappedTo($c_i$) then
                $c_i$_Cluster_total_weight+=$sim_{Cosine}(w_i, c_i)$;
        $Cl_{List}$_Update($c_i$, Cluster_total_weight);
    $Cl_{List}$_Sort by Cluster_total_weight DESC;
    // (4) Assigning the common concepts to the web pages;
    foreach $c_i \in Cl_{List}$ do
        foreach $w_i \in Can_{List}$ do
            if $w_i$_Contains($c_i$) then
                if $F_{list}$_Does_not_Contain($w_i$) then
                    $F_{list}$_Add($w_i$,$c_i$);
```

Figure 4: The 3C algorithm.

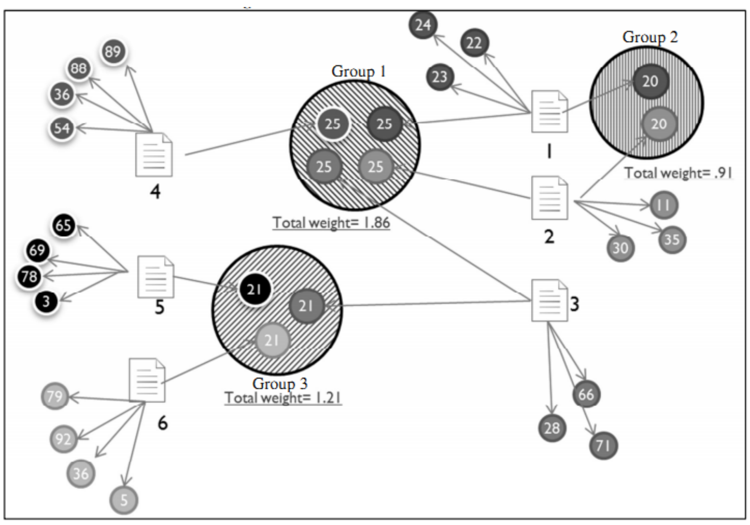


Figure 5: An example of the 3C algorithm.

- A concept may occur over a long period and its occurrence is high. This concept should be considered as a short-term interest for a start, and once it is confirmed with time, it should become a long-term interest. For instance, a new Computer Science student will become interested in topics related to his subject. These topics should be treated as long-term interests.
- A concept may occur in a short or long period, and its occurrence is very high. Then, this concept disappears, but after a while the user shows interest in this concept again. Such behaviour is difficult to recognize as a user might shift or drift his interests to new or urgent ones, and once he is done with these new interests, he returns to the previous ones. An example would be when a user is interested in reading topics related to his work, but if he decided to take a holiday and travel abroad his interests would probably shift to new topics related to the visiting country. However, once back from the holiday, the user would likely return to his earlier work-related topics.

We make use of these heuristics in our learning and adaptation mechanism and to this end we introduce a multi-agent system which deals with the complex processes of recognizing, adding, updating and deleting user interests. We make use of a multi-agent system architecture for two main reasons. Firstly, the use of agents with specific responsibilities within the learning and adaptation process enables us to delineate functionalities and interactions between them clearly; the flow and processing of information are clearly defined. This also facilitates the use and processing of the same piece of information from multiple perspectives as different agents in the system may have access to the same information, but they may be viewing it from a different perspective and dealing with it in a different way. The second main reason is that of flexibility and extensibility. The various agents within the system can be upgraded and modified on their own without affecting the rest of the agents. Hence, we can easily make changes to the behaviour of the agents and their underlying algorithms in the future separately. This also enables us to test the system in a more systematic way and reconfigure algorithms in isolation and in conjunction with each other.

The multi-agent system handles the users' browsing behaviour in general, and the change in their interests in particular and it consists of three layers to provide dynamic learning and adaptation: the session based, short-term and long-term layer (Figure 6). Each layer consists of one or more agents that are responsible for a (set of) task(s).

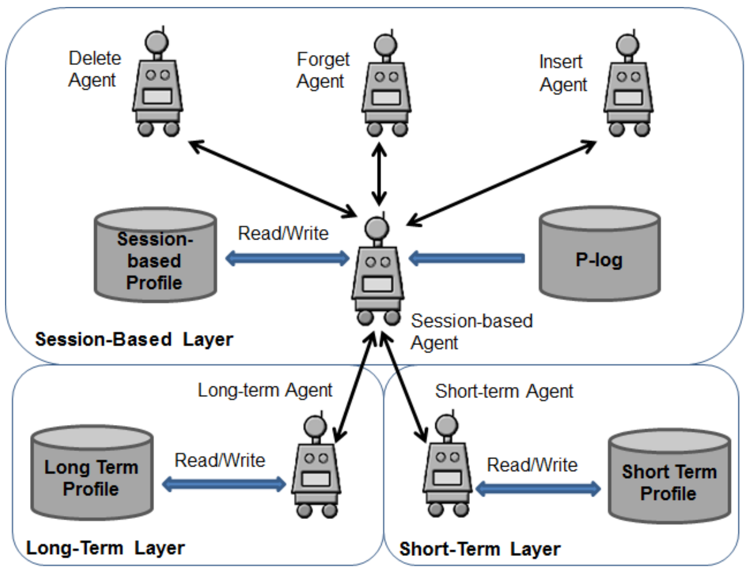


Figure 6: A multi-agent system for modelling dynamic user profiles.

3.2.1. Session-Based Layer

In order to learn user profiles, we need to observe, process and then learn from user browsing behaviours for each session, hence this process is activated after every session. The learning and the adaptation processes consist of adding, forgetting and deleting concepts from a user profile, computing the interest weights for the visited concepts and preparing a list of candidate concepts to be analyzed by the short-term and long-term layers. This layer consists of a number of agents each responsible for one or more tasks.

Session-based Agent: This is a key agent in the multi-agent system as it is responsible for four main tasks: (i) collecting data from the P-log file; (ii) communicating with other agents to calculate the latest interest weight for all collected concepts from the P-log; (iii) storing all the processed concepts and their interest weights in a session-based profile (SBP); (iv) communicating with the Short-term and Long-term Agents to enable them to discover short-term and long-term interests respectively.

Once a session finishes, the data in the P-log file are processed and stored in a SBP. One critical difference between the P-log and the SBP is that the P-log consists of URLs, timestamps, durations and concepts that represent these URLs, while in the SBP there are no URLs but just concepts and their associated attributes. These attributes include: (i) The *Status* which can be: positive status such as *Browsed-concept* or *Confirmed-concept*, or negative status such as *Forgotten-concept* or *Deleted-concept*. (ii) The *Relevance_size* which refers to how much a concept is relevant to a user interest. The relevance size can be measured based on user feedback about each concept. If the user feedback is positive, then the relevance size increases, but if it is negative, then it decreases. (iii) The third attribute is the *Frecency* which represents the interest weight that indicates how much a user is interested in a concept. (iv) Finally, the *Frequency* attribute represents how many URLs from the P-log file have been mapped to a particular concept.

The Session-based Agent starts by extracting from the P-log file the concepts that represent a web page, the *Status* and *Frequency* attributes. Next it communicates with the other agents in order to set the *Relevance_size* and compute the *Frecency* weight for each concept. We borrow the term frecency, which is a combination of frequency and recency, from the field of web browsing to represent the weight of an interest (MozillaWiki, 2010). This process is performed daily and hence all new interests as well as existing ones are processed daily to adapt them to any change in user behaviour.

Insert Agent: This agent is responsible for processing all concepts with positive status that are received from the Session-based Agent. The positive status may be associated with different events where each event has different value to represent its importance. The different events in the system (and depending on the domain of application these may vary), need to be assigned numerical values in order to illustrate their difference and their relative importance. We identify two events: *Browsed-concept* and *Confirmed-concept*. The *Browsed-concept* event is the default one that is assigned to any concept that is browsed by a user. If such a concept appears in two subsequent sessions, it is assigned the *Confirmed-concept* event. This event has a higher weight than the *Browsed-concept* one, as concepts appearing in more than one session would likely be of more interest to users than those that appear in just one. Other events could be added based on the domain of application, such as *Purchased-item*, *Printed-concept* or *Bookmarked-concept*.

Whenever a web page is browsed that has a classified concept, the interest weight (i.e. frecency) for that concept is accumulated using equation 4. The frecency value is computed based on two factors: the event and the duration that are associated with the web page. For instance, if the concept has been browsed, the event weight is assigned to be 100, while if it is a *Confirmed-concept*, then we assign a weight of 150. We could have assigned different values for the weights at this stage (e.g. 1 and 1.5 respectively); the important thing is to be able to differentiate between these two events and to reflect the difference in their importance. The second factor, the duration, is the time spent on a web page and as it is in seconds, it is normalized. We consider the duration when computing the interest weight for a concept since time spent on web pages has been shown by studies including (Konstan et al., 1997; Claypool et al., 2001) to be a good indicator of user interest.

$$Fre = \sum_{c_i \in C}^{n} \left(\frac{c_i.k}{100} \right) * E.weight;\ E.weight = \left\{ \begin{array}{ll} 100 & \textit{if Browsed-concept} \\ 150 & \textit{if Confirmed-concept} \end{array} \right\} \quad (4)$$

Where $c_i.k$ is the time a user spends reading c_i, $E.weight$ is a weight that is added based on the event of the concept. The interest value of a concept c_i is computed by summing up all the frecency values for each web page mapped to this concept. After assigning a frecency weight to each new concept, the *Insert Agent* adds this concept and its properties to the SBP.

Forget Agent: This agent handles the behaviour that occurs when a user loses interest in a concept. An effective personalization system should be able to adapt to this change in user behaviour by forgetting such an interest from the user profile. However, such a concept should not be deleted immediately, but instead it should be forgotten gradually until the concept is confirmed as no longer being of interest to the user. Of course, not all interests are forgotten with the same pace. To account for this, the pace of the forgetting process depends on three factors. The first is the *Relevance_size* that is associated with each concept and is an indicator of the user's strength of interest. The $Relevance\_size$ is increased/decreased depending on whether a user shows positive feedback towards a concept (i.e. by either searching for or browsing web pages related to this concept) or negative. The second factor is the recency of a concept as old concepts are forgotten faster than new ones. The final factor is related to the introduction of new interests to a user profile. If a user has started to lose his/her interest in a concept, and at the same time started to acquire new interests, then this behaviour might indicate that there is a drift in the user interests.

In related work, and in order to encode the assumption that the user's preferences gradually decay as time passes, Sugiyama et al. (2004) introduced a forgetting factor for interests in the user's profile:

$$e^{-\frac{log2}{hl}(d-d_{t_{init}})} \quad (5)$$

Where $d_{t_{init}}$ is the day when term t occurs initially and d is the number of days after the occurrence on day $d_{t_{init}}$. The parameter hl is used to indicated the half-life span and is set to 7 under the assumption that user interests reduce to $1/2$ in a week (Sugiyama et al., 2004).

In order to apply a forgetting process to our profiles, we borrow the idea of the time-based forgetting factor from (Sugiyama et al., 2004). In contrast to using a predefined half-life span, our forgetting factor is more dynamic in order to account for the three aforementioned factors and hence the new frecency of a concept (New_Fre) is calculated as follows:

$$c_i.New\_Fre = c_i.Old\_Fre * e^{-\frac{log2}{(c_i.Relevance\_size*2)}(G_m - G_l) + (\#new\_interests : d_t)} \tag{6}$$

Where $c_i.Relevance\_size$ is the relevance size of a concept c_i, G_m is today's day, G_l is the day of the last occurrence and # $new\_interests$ is the number of new interests that are introduced in the G_m. Unlike (Sugiyama et al., 2004) who divided log2 by a predefined half time, in this equation we divide log2 by $(c.Relevance\_size * 2)$ and use the recency and the number of new concepts, so the larger the value of these elements, the faster the forgetting process. This makes our forgetting factor more dynamic and we take into account individual user behaviour as users, for instance, may acquire new interests at different rates.

In Equation 6, we rely on the concept's $Relevance\_size$, recency and the number of new concepts introduced to the user profile to handle the user's interest change and interest drift. As explained in (Godoy and Amandi, 2009), interest drift occurs when an interest experiences a gradual change. The proposed mechanism handles such behaviour by initializing the process once a user shows no more interest in a concept. The pace of forgetting further depends on whether a user has got interested in new concepts or not. The more new concepts in the SBP, the faster the forgetting process. This is particularly important in coping with interest drift as the introduction of new concepts at the time of losing interest in older concepts indicates implicitly that there is a drift in the user interests. Hence, older concepts should be forgotten and replaced by new ones.

Finally, it is important to note that the forgetting process for a concept might be paused when a user shows interest again. The $Relevance\_size$ would also increase to reflect this change. But if a user continues not to show interest in the concept, this will be gradually forgotten until the end of the $Relevance\_size$ (the size of zero). Then this concept is sent to the *Delete Agent*.

Delete Agent: This agent manages the gradual deletion of a concept from a user profile. The difference in the operation of the *Forget* and *Delete Agents* is that a concept with a *Forgotten* status can still be an interesting concept to a user and appear in his/her profile, while a concept with a *Deleted* status means that this concept is no longer interesting and it should not appear in the profile. When a concept is passed on to the *Delete Agent*, this is removed much faster based on the time of the last appearance of the concept and until the weight reaches a predefined threshold and then it is removed altogether. We use the next equation to compute the new frecency of a *Deleted* concept:

$$c_i.New\_Fre = \left(\frac{c_i.Old\_Fre}{G_m - G_l} \right) \tag{7}$$

Where G_m is the current date and G_l is the date of the last occurrence of c_i. Hence, if there has been no interest shown for long periods of time, the frecency will be reduced accordingly.

Once all concepts are processed, the Session-based Agent stores the final results into the SBP. Subsequently these are used to discover short-term and long-term user interests, therefore, the last task of the Session-based Agent is to communicate with the Short-term and Long-term Agents.

In this work, we distinguish between long-term and short-term interests as they are different in nature. Discovering the short-term interests helps a personalization system to shed light on users' current interests which are likely to change frequently. To illustrate, a user might be interested in a holiday in France. After returning from the holiday, the user would most likely lose interest in France and therefore this should constitute a short-term interest. A typical personalization system would provide relative personalized contents to this user that match his/her interest in France. The long-term interests on the other hand, can play an important role in providing more effective personalization services to users that match their long-term preferences. If the same user has a long-term interest in science fiction movies, this can be combined with the short-term one by suggesting new science fiction movies and cinemas while in France.

However, the process of discovering short and long-term interests presents challenges. The main one arises from the question of how long (short) is *long* (*short*). Users may exhibit very different browsing behaviours: some may change their interests frequently, while others may rarely change. Therefore, identifying length/duration is subjective to each user. We address this issue in our work by incorporating long and short-term layers that consist of adaptable mechanisms to discover short and long-term interests specific to each user.

3.2.2. Short-term Layer

Our goal in this section is to develop a new mechanism to learn the user's short-term interests and to be able to adapt to the various browsing behaviours listed in the beginning of section 3.2. This layer includes two components: the short-term profile (STP) that is used for storing all the concepts identified as short-term interests, and the short-term agent (STA) that is responsible for tasks such as discovering, maintaining and storing short-term interests in the STP.

The STA discovers short-term interests by examining the concepts stored in the SBP. As each concept in the SBP is associated with a frecency that represents its importance, the STA uses these values to determine which of these concepts are short-term interests. One way to discover short-term interests is to pre-define a threshold so all concepts with weights above it are selected as short-term interests. This method has been adopted by studies such as (Li et al., 2007) and (Grcar et al., 2005). However, users are different and using the same threshold for every user would very likely result in modelling inaccurate interests.

We overcome this problem by developing a self-tuning mechanism to calculate a threshold which is adaptable to different users as well as to changes in each user's browsing behaviour. To identify this threshold, we first observe the frecency average of a user's browsing behaviour for each session s. Normally, when browsing, the user would navigate to both interesting and uninteresting pages. The uninteresting topics would be assigned low frecency weights, while interesting ones would be assigned high weights (i.e. based on the time spent on each page). The threshold is computed as the ratio of the total frecency values for each visited concept in each session to the total number of concepts that have been visited in all sessions:

$$Threshold = \frac{\sum_{all\ s \in u_i}^{n} Frecency\_values}{\mid Total\ number\ of\ concepts \mid} \tag{8}$$

Hence, all the concepts in the SBP that have weights above this threshold are stored in the STP. As this threshold adapts to various browsing behaviours, each user would have a different threshold.

Relying solely on a threshold is not adequate, as we also need to control the size of the STP. This is because users might be interested in a large number of topics on one day, but on another they may have less interesting topics or even no interests at all. Unlike some studies such as (Li et al., 2007) and (Grcar et al., 2005) which rely on fixed size profiles to store interests, in this layer we propose a mechanism to adjust the STP size by either expanding or limiting it to reflect the actual user interests. Taking into account the user's browsing behaviour, we compare the average of the current session to the average of the last session and if it is larger/smaller, then the size of the STP is increased/decreased accordingly, otherwise it remains the same. We use the average of each session as it provides an indication of a user's behaviour. To prevent the STP size from being increased indefinitely or diminished we set a maximum/minimum STP size which can be identified based on the domain of application at hand. The frecency average for a session is computed by:

$$Fre\_average = \frac{\sum_{c \in s_i}^{n} Frecency\_values}{\mid Total\ number\ of\ concepts \mid} \tag{9}$$

Where $\sum_{c \in s_i}^{n} Frecency\_values$ is the total frecency values for all concepts c that have been visited during session s_i.

Finally, the STA also needs to be able to deal with the problem of interest shift. An interest shift can be defined as abrupt change in user interests (Godoy and Amandi, 2009). Unlike interest drift which is a gradual interest change and which has been dealt with by the *Forget Agent* in section 3.2.1, the interest shift is more difficult to recognize as it happens suddenly (Gorgoglione et al., 2006). Users can sometimes become suddenly interested in some topics and then once these topics lose their importance, they simply abandon them. For instance, when the Football World Cup is about to start, sports fans would become interested in this and might not exhibit interest in other sports. But when the event finishes, they would likely shift their interest to other sports, or new events. As it is not easy to recognize such shifts at the time of their occurrence, personalization systems can suffer from lower performance.

The STA handles interest shift by adding any new concept with a frecency weight above the computed threshold to the STP. However, if the STP is already full, then the concept would not be added. To deal with this problem, the STA replaces the concept that has the lowest frecency weight in the STP with this new concept, even if this concept has a lower frecency weight than the replaced one. We attempt to replace new concepts with existing concepts in the STP when it is full in order to discover a user's interest shift. If this new concept appears again in the subsequent session, the *Insert Agent* in the session-based layer would process it and it would be treated similarly to the other concepts, while if the concept does not get confirmed, then it will be forgotten. In this case, our system would recognize and handle the interest shift when it happens. We call this the *Replacement-task* as older concepts are replaced by new ones.

3.2.3. Long-term Layer

The Long-term layer is responsible for learning, recognizing and storing long-term interests that are confirmed with time and consists of the long-term profile (LTP), which stores all interests that are recognized as long-term ones, and the long-term agent (LTA), whose task is to recognize, maintain and store long-term interests in the LTP.

Long-term interests are different in nature from short-term ones as they do not change as frequently, but instead they appear more regularly over a longer period in a user profile. For instance, users who work as programmers might have a long-term interest in programming languages which might appear regularly in their profiles. Unlike the short-term interests which rely mostly on recency when computing their interest weights, the long-term interests should rely more on the frequency of occurrence. The LTA computes the frequency weights for each concept in the SBP using the following equation:

$$c_i.FW = (N_{occ} * N_{day}) * \left(\frac{G_m - G_f}{P.log_{\,age}} \right) \tag{10}$$

Where $c_i.FW$ is the frequency weight of a concept c_i, N_{occ} is the number of times the concept c_i occurs in the P-log file, N_{day} is the number of days that the concept c_i occurs in, G_m is the day when the process is launched (i.e. today's day), G_f is the day of the first appearance of the concept c_i and finally, the $P.log_{age}$ is the age of the P-log. By using the age of the P-log we can detect which concepts have been appearing for longer. In this equation, the more frequent occurrences of a concept, the higher the weight assigned to it. Moreover, the interests that occur over a longer period receive higher weights than those that occur in a short period as they seem to be more stable. Unlike the STA which computes the short-term interests' frecency after each session, the LTA computes the frequency weights for long-term interests periodically such as once or twice a month based on the domain of application. As long-term interests do not change as frequently as short-term ones, there is no need to compute them as frequently.

Once the frequency weights for all concepts in the SBP are computed, the LTA needs to identify the interests that should be treated as long-term interests. We introduce a new mechanism to compute a threshold so that all concepts above it are stored in the LTP. Similar to the short-term layer, we propose a dynamic threshold that is based on each user's browsing behaviour. We first compute the frequency for each concept using equation 10 and then we compute the Standard Deviation of all the concepts in the SBP:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(ci.FW - AV)^2} \tag{11}$$

Where σ is the Standard Deviation, N is the total number of concepts, $c_i.FW$ is the frequency weight of a concept c_i, and AV is the average of all the frequency values for all concepts. The threshold is then:

$$Threshold = \sigma + \left(\frac{\sum_{i=1}^{N} c_i.FW}{\mid N \mid}\right) \tag{12}$$

Where $\sum_{i=1}^{N} c_i.FW$ is the total of all the frequency weights for all concepts. Finally, all the concepts with frequency weights higher than the computed threshold would be stored in the LTP. Again, instead of predetermining a threshold and fixing it for all users at the same level, we utilize the information on the frequency weights of the user's interests and their standard deviation and this enables us to tailor the threshold to the specific user's pattern of behaviour.

Long-term interests are more stable than short-term ones, so the process of discovering them is much easier. It is not appropriate to use the same technique to discover the short-term interests as they are unstable and change frequently, and they are also subject to interest-shift or drift. When finding the short-term interests, we should not use the standard deviation with the frecency average as a threshold as we do not want to limit the threshold to select just the outstanding concepts, but indeed we need to select all the concepts with frecency weights larger than the frecency average. This is because, when a user changes his/her interest, the new interest would likely have a low frecency weight in the beginning, so using just the frecency average would allow us to capture such interests. However, in order to deal with the problem of selecting too many short-term interests, in the short-term layer, we limit the size of the STP by setting a maximum number of interests. So for this particular reason, we do not have to set a specific size for the long-term layer as using the long-term threshold would likely limit the long-term interests to only those most frequent and outstanding ones.

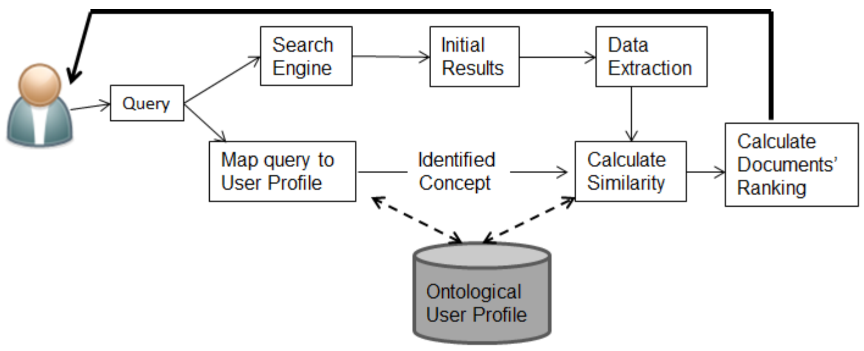


Figure 7: The re-ranking process for the personalized web search system.

3.3. Personalization System Phase

We integrate the work described so far with an experimental web search personalization system in the third and final phase. As part of this, we also introduce an experimental re-ranking approach. We aim to demonstrate that our approach on dynamic user profiles can effectively capture user interests as well as the change in user behaviours including interest shift and drift and can be used as part of a personalization system to provide more effective and accurate services to users. However, the methods that we have developed are generic enough and can be integrated and utilized by diverse personalization systems.

In the proposed system, which is illustrated in Figure 7, when a user submits a query, two processes take place simultaneously:

1. The query is mapped to the ontological user profile to identify the most similar concept that represents it. For this reason, the cosine similarity is used to compute the similarity between a query and all the documents for all concepts in the user profile. We also pre-identify a threshold, so the highest similar concept which is above this threshold will be selected to represent this query.
2. The query is passed to any search engine (e.g. Google or Yahoo) in order to retrieve initial search results. The contents of each retrieved result are then extracted and stored in a separate document using various text analysis techniques such as the ones that have been used in section 3.1.1.

We subsequently compute the similarity between each retrieved search document and the contents of the interesting concept that represents the initial user query. Finally, we re-rank all the retrieved search results in descending order and present them the user. The ranking approach is described in Figure 8.

4. Evaluation

In general, the evaluation of recommender, web personalization or systems that deploy user profiles is recognized to be difficult and expensive (Yang et al., 2005). As such systems may have different purposes and they tend to be very complex, making direct comparisons is difficult. The evaluation strategies commonly used can be divided into three main categories (Shani and Gunawardana, 2011): offline, user-centered studies and online evaluations.

In offline evaluations, existing real datasets or artificial datasets are employed to assess the performance of the system. The use of such datasets minimizes the cost of the evaluation and can facilitate reuse. Although there are datasets such as the MovieLens[5] one that can be used for collaborative filtering systems, rich user datasets that can be used for personalization systems are almost non-existent.

[5]http://MovieLens.umn.edu

```
Algorithm 3: The Re-ranking Model
$C = \{c_{i.1}, ..., c_{i.n}\}$ Concepts in a user ontological profile $UOP$.
$q$= is a query that is submitted by a user.
$ISR = \{r_i, ..., r_n\}$ is the initial search results of a query $q$.
$O$= is the reference ontology which consists of concepts $C$ and documents $D$ that represent
these concepts.
Input: Ontological user profiles $UOP$ that consists of short-term and long-term interests
       which are linked to web pages), and the submitted query $q$.
Output: Re-ranked search results.
begin
    // (1) Mapping the user query to a reference ontology to find the highest similar concept;
    foreach $c_i \in O$ do
        Calculate sim($c_i$_document, q_textual-contents);
        $Max\_sim = 0$;
        Found-concept=0;
        if $Max\_sim <$ sim($c_i$_document, q_textual-contents) then
            $Max\_sim$ = sim($c_i$_document, q_textual-contents);
            Found-concept= $c_i$;
    // (2) Re-ranking search results based on the similarity between each result and concept's
    contents in a user profile;
    if $Found - concept \neq null$ then
        Initialize Re-ranked-List;
        foreach $r_i \in ISR$ do
            Calculate sim(Found-concept_document, $r_i$_contents);
            Re-ranked-List_Add($r_i$, sim(Found-concept_document, $r_i$_contents));
        Re-ranked-List_Sort(by sim);
        Show all results based on the Re-ranked-List weights.
```

Figure 8: The re-ranking algorithm.

In user-centered evaluations, real users are recruited in order to collate realistic behaviours to test the performance of the system (Shani and Gunawardana, 2011). Such evaluations involve three steps: (i) recruiting users; (ii) creating a set of tasks to be completed by the users; and (iii) collecting and analyzing user interactions and behaviours. Such evaluations are expensive to design and carry out and inevitably tend to be limited in scale as it is very difficult and costly to recruit large numbers of users. In particular, the tasks need to be carefully drawn so that they are able to provide appropriate and relevant results. To be able to draw meaningful comparisons among systems that are being tested, users need to be able to repeat the same or very similar tasks, hence these usually take the form of *simulated work tasks*. Borlund (2003) made a number of recommendations when creating tasks for such evaluations. User evaluations are able to provide a better indication of the effectiveness, accuracy and efficiency of personalization systems than using artificial datasets as systems are tested in realistic settings. They can also provide answers to a wide range of questions directly by the users and therefore afford a more well-rounded evaluation.

In online evaluations, a system is essentially deployed and evaluated in a real environment. This is considered to be the best way to evaluate any system as in a real environment a system can be thoroughly tested against real users' behaviours (Montaner, 2001; Shani and Gunawardana, 2011). However, online evaluations present a number of challenges. A system needs to be sufficiently well developed to make it available to real users. This is expensive in terms of time and effort. The experimental algorithms deployed may have a low performance, and hence, exposing them as part of an online evaluation may affect the user experience and potentially if the evaluation is carried out by a company, the users' trust in a company, product etc. may be negatively affected.

To evaluate our work, we have undertaken a user-centered evaluation that enabled us to collect real user data which we subsequently analyzed to examine the performance of the various aspects of our work. We have split the evaluation into two parts (i) evaluation of the mapping and user profile modelling methods; (ii) evaluation of the methods within a search personalization system.

5. Evaluation I: Mapping & User Profile Modelling Methods

In setting up our user-centered study, we chose the domain of computers as the application domain. Our methods require the use of a reference ontology. Though we do not make any assumptions on the ontology used, for the purposes of this study, we first created a reference ontology using information extracted from the computer category from the Open Directory Project (ODP) Ontology. We used the ODP as it is considered to be the largest manually constructed directory consisting of websites categorized in related topics. The computer directory contains more than 110,000 websites categorized in more than 7,000 categories. In order to train the classifier, all the websites under each category were fetched. Furthermore, all the contents of all websites were extracted and combined in one document. Hence, each category $c_i \in O$ is represented by one document d_i that includes textual contents tx from web pages W associated with the c_i. All the non-semantically related classes (e.g. alphabetical order) were removed from this ontology. This resulted in a total of 4,116 categories and about 100,000 training websites whose contents were extracted and combined in 4,116 documents in our reference ontology. For each document, we apply dimensionality reduction techniques as explained in section 3.1.2 to minimize the number of terms. After applying these techniques, a vector space model is used to convert the terms in these documents to vectors. For this purpose, the TF-IDF classifier is used which gives each term $t_j \in d_i$ a weight from 0 to 1 using equation 1.

Next, five scenarios were created to address different user browsing behaviours with respect to the heuristics that were discussed in section 3.2. The aim of these scenarios is to test the learning and adaptation ensuing from our methods when different behaviours occur and in particular interest shift or drift. For each scenario, a set of tasks were created to simulate an environment where users would interact with the system as they perform their own tasks. We selected 35 random topics from the computer ontology (e.g. XML, C#, Computer security) as the basis for the experimental tasks. A total of 90 unique tasks and sub-tasks were created based on these 35 topics. The tasks take the form of finding a specific piece of information, or writing a short paragraph. Finally, 30 participants with a computer science background were invited to participate in our experiment during a 20 day period. The participants were divided so that each scenario was undertaken by six participants. These scenarios are described in more detail in appendix A.

To track user browsing behaviour, a Firefox browser was used with a modified add-on component called Meetimer[6]. Meetimer was modified to implicitly track user browsing behaviour including all the visited web pages, timestamps and durations. An SQLite database was used to store all the user sessions.

After 20 days, 30 log files from 30 different users were collected. These 30 users together surfed 9,360 web pages, and an average of 15.6 web pages a day for each user. We then processed the collected data to create processed log files (P-log files) by fetching all the visited web pages, and extracting all their contents. Moreover, we removed all the HTML tags and other noise data such as advertisements. All the stop words were removed and then the Porter stemming algorithm was applied (Porter, 1997) to reduce the terms' dimensionality. In the next stage, we analyzed the collected data by first examining the performance of the GEW and 3C algorithms, and then the performance of the learning and adaptation processes.

5.1. *Evaluation of the GEW and 3C Algorithms*

To evaluate the proposed mapping algorithms GEW and 3C, we conducted three experiments to analyze different aspects that impact on their overall performance.

Tuning of the GEW Algorithm. In this experiment, we examine the best values for the parameters α and γ. The former controls how much weight is transferred from one concept to its upper-concept. The latter refers to the number of highest similar concepts to a web page from the reference ontology that GEW would be applied on. Determining the best γ values allows us to remove the concepts that add no value when applying the GEW algorithm. We have experimented with $\alpha = \{0.1, ..., 1\}$ and $\gamma = \{5, 10, 20, 50, 70, 100, n\}$ where n refers to considering all concepts when applying the GEW. We examine all of these settings to measure the accuracy of mapping the user visited web pages to a reference ontology using equation 13. The accuracy is computed as the ratio of the total number of positively mapped web pages to correct concepts from a reference ontology to the total number of web pages visited by all users.

$$Accuracy = \frac{|\ Positive\ mapped\ web\ pages\ |}{|\ All\ web\ pages\ |} \tag{13}$$

Figure 9 shows the accuracy percentages for applying the GEW on all the visited web pages by all users in all scenarios using different α and γ values. It can be clearly seen that the accuracy of applying the GEW algorithm to all the similar concepts is relatively low, and the accuracy increases whenever γ decreases. In particular, when applying the GEW to the top 5, 10 and 20 concepts, the accuracy of the mapping process shows considerable improvement. Surprisingly, the best accuracy result is achieved when $\gamma = 10$ but not when $\gamma = 5$. A possible explanation for this is that the top 10 similar results to a web page could hold the most important concepts that are likely to be related to this page, while considering just the top 5 results may miss some very important concepts. When it comes to α, interestingly the accuracy results improve when α is close to 0.5, and clearly tends to decrease when α tends to the maximum or minimum. This is because when $\alpha = 1$, most of the inaccurately mapped concepts are too general to represent user interests, while

[6]https://addons.mozilla.org/en-US/firefox/addon/meetimer/

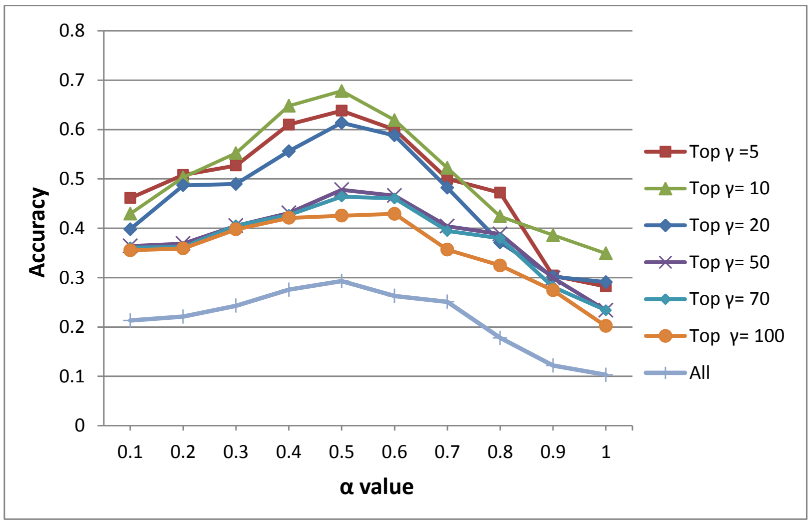


Figure 9: The accuracy results of applying the GEW algorithm on all web pages visited by all users with different α and γ values.

β value	5	10	20	30	40
Accuracy	0.7805	0.7448	0.731	0.7108	0.669

Table 1: Accuracy results of the 3C algorithm with different β values.

when $\alpha = 0.1$ most of web pages were mapped to leaf nodes from the reference ontology which happen to be too specific to represent these interests. For $\alpha = 0.5$, a balance between general and specific interests is maintained. Based on these results, we assign $\alpha = 0.5$ and $\gamma = 10$ for all subsequent experiments.

Tuning the 3C Algorithm. To apply the 3C algorithm, we need to set β which acts as a threshold and determines the number of concepts that are used as candidates. To identify the best β value, we apply the 3C algorithm to the top 40, 30, 20, 10 and 5 concepts. For each value, the 3C algorithm attempts to cluster web pages with common candidate concepts together in order to identify the most accurate concept that correctly represents them. Table 1 shows the accuracy results using the measure in equation 13.

Surprisingly, it can be seen from Table 1 that all thresholds have achieved very close accuracy results. The top 5 threshold achieved the highest accuracy result of 0.7805, while the top 10 and top 20 performed slightly worse with 0.7448 and 0.731 respectively. This result may be interpreted as suggesting that the most appropriate concept that should be mapped to a web page is likely to be among the top 5 most similar concepts to a web page. Finally, these results support our hypothesis that mapping web pages as groups can enhance the mapping process as the accuracy of applying just the GEW is 0.6779 while by using both the GEW and 3C algorithms, the accuracy increased to 0.7805. Based on these results, we assign $\beta = 5$ in all subsequent experiments.

Comparing Against Other Mapping Techniques. So far, we have examined different values to find the best possible configuration that can enhance the GEW and 3C performance. However, it is essential to examine how these algorithms perform against other techniques in the literature. Hence, in this experiment, we compare our mapping algorithms (GEW and 3C) to three different mapping techniques in the literature. The first is the original cosine similarity (OCS) which computes the similarity between each collected URL and documents in the reference ontology. Each URL is then represented by the highest similarity

Method	Accuracy
GEW & 3C	0.7805
OCS	0.4489
Adding 50%	0.6158
SAS	0.4223

Table 2: Comparison of four different mapping approaches.

concept from the ODP. The second technique, which was suggested by (Middleton et al., 2004) and (Kim et al., 2007) is adding 50% of each sub-concept's weight to its super-concept, and repeats this process until reaching the root. The third technique is the Sub-class Aggregation Scheme (SAS) that was proposed in (Daoud et al., 2008) where user interests are represented using level three of the ontology. The weight for each level-three concept is computed by adding the weights of its sub-concepts. In this experiment, each visited web page for each user is mapped by applying all four techniques. Table 2 shows the overall average accuracy for each mapping technique for all the users.

According to Table 2, the OCS and SAS performed poorly as the former achieved an accuracy of 0.4489 while the latter a slightly lower result at just 0.4223. It is somewhat surprising that the OCS performed slightly better than the SAS. This poor performance of the SAS could be attributed to that all the visited web pages are mapped only to concepts in level three of the reference ontology. This causes mapping these web pages to concepts which are too general to represent them. On the other hand, adding 50% of each sub-class to its super-class shows considerable improvement in the mapping with accuracy 0.6158. This could be due to the main underlying assumption that if a user is interested in any concept, then s/he is likely have some interest in its super-concept. Although this technique improved the overall mapping accuracy, the percentage which is added to the super-classes is very high (50%). As a result, 44% of all the incorrectly mapped web pages were mapped to level 1 and 2 super-concepts that are too general and broad to represent user interests. The overall accuracy of the GEW and 3C algorithms shows a marked improvement over all other techniques, achieving an accuracy result of 0.7805. This improvement shows that GEW and 3C can overcome some of the drawbacks of other techniques as well as keep a balance between the general and the more specific interests.

5.2. *Evaluation of the Dynamic User Profiling Methods*

Here we aim to evaluate the performance of the proposed dynamic user profile and the effectiveness of the learning and adaptation processes. Hence, we first evaluate different features of our system, and then compare its overall performance against other modelling approaches in the literature. For all of these evaluations, we measure the precision of learning and adapting the user profiles for each day of the experiment duration (20 days) using equation 14:

$$Precision = \frac{\mid Number\ of\ correct\,learned\,and\ adapted\ interests\mid}{\mid Total\ number\ of\ actual\ interests\mid} \tag{14}$$

The precision in this equation is computed as the ratio of correctly modelled interests in user profiles to the actual number of interests in the scenarios.

Modelling Dynamic User Profiles. First, we examine the effectiveness of the *Replacement-task* feature which was proposed in section 3.2.2, and aimed at dealing with the interest-shift behaviour that might occur

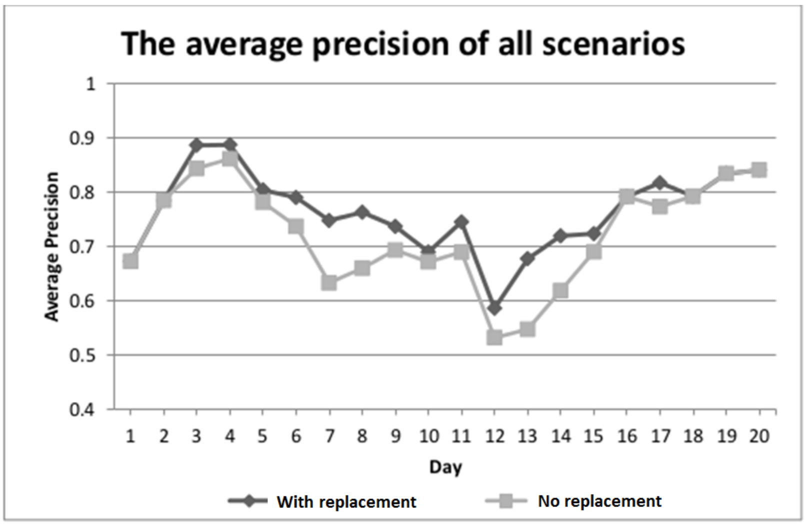


Figure 10: Average precision for each user over the 20 day period.

when a user browses the Internet. We compare how our system performs when the *Replacement-task* is enabled and when it is disabled for each day of the experiment and for each user profile. As Figure 10 illustrates when the feature is enabled the average precision for all user profiles is 0.7649, while when it is disabled it drops slightly to 0.7228. In more detail, we can see that in some days such as days 5 and 11, a sudden drop in the precision is observed when the *Replacement-task* is enabled as well as when it is disabled. These drops were somehow expected as in these days, the user profiles experienced a shift of interests. Interestingly, when the *Replacement-task* is enabled, the precision results during the interest-shifts are considerably better than when it is disabled. Moreover, after each interest-shift such as after days 5 and 11, the system with the feature enabled managed to effectively learn and adapt to the changes in user behaviours quickly as the precision consistently improved. These results show that using the *Replacement-task* feature can detect and handle the interest-shift quite well.

Next, we evaluate the performance of different threshold techniques that are used to discover short-term and long-term interests. First, we evaluate three such techniques for discovering short-term interests:

- **Fixed number of interests (Fixed)**: A fixed number of topics in the SBP is selected and considered as short-term interests. This threshold technique has been applied by studies such as (Grcar et al., 2005) and (Li et al., 2007). In this experiment, we select just the top 5 topics from the SBP as it was suggested in (Grcar et al., 2005).
- **Largest gap (LGap)**: In this technique, we first find the largest gap among the frecency values that are associated with the topics in the SBP, and then consider all the topics with frecency values larger than the largest gap as short-term interests.
- **Our approach**: We employ the system described with all the aforementioned features and use the average frecency and the *Replacement-task* feature to select the correct short-term interests for a user.

Figure 11 shows the overall comparison between the above three threshold techniques over 20 days. The LGap technique achieved the lowest precision performance at an average of 0.649. A possible explanation for this is that when a new interesting concept is browsed by a user, its interest weight would not be high.

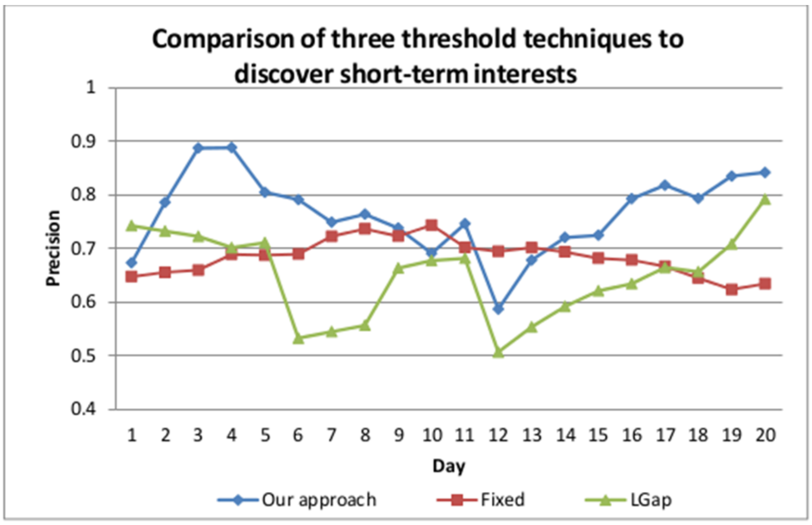


Figure 11: Comparison of three techniques for discovering short-term interests.

As a result when the largest gap is computed, only the existing interests with high interest weights are considered as short-term interests, while the new browsed interests are not (as their interest weights are likely to be less than the largest gap value). In addition, when using LGap, sharp drops in the precision can be observed after days 5 and 11. These drops occur because of the interest-shift and drift in users' interests in the scenarios that have been tested which cannot be captured by LGap. When it comes to the Fixed number of interests threshold, surprisingly the performance is slightly better than LGap with an average precision of 0.683. What is interesting, is that this technique is not affected by the interest-shift and drift that occur in the tested scenarios. In days 10, 12 and 13 this technique managed to achieve the best precision in comparison to the others which shows that it can handle all the interest-shifts and drifts quite well. Since the five most recent concepts are considered as short-term interests on each day, regardless of their interest weights, it manages to capture the interest-shift and drift; when these occur, the new interests are simply added to the user short-term profiles. Although this technique has no problem with the interest-shift and drift behaviours, it has two limitations that affect its overall precision. The first limitation is that the size of the short-term profile is limited to just five interests per session. Hence, if a user has more than five interests at the same time, this technique would eliminate and ignore some of the interesting concepts. The second limitation is that the interest weights for concepts are not considered when selecting short-term interests. As a result, any concept even uninteresting ones would be considered as interesting to a user by this technique. The last threshold technique is our approach of using the average frecency with the *Replacement-task* feature. As it can be seen from Figure 11, our technique achieves an average precision of 0.764 which is a significant improvement on the other techniques. This result demonstrates that our approach can provide an effective mechanism to learn and adapt to the changes in user browsing behaviours.

When it comes to discovering long-term interests, we evaluate and compare the performance of the following four techniques:

- **Fixed number of interests (Fixed)**: We apply the threshold technique in (Grcar et al., 2005) where the short-term profile stores the most recent visited pages, while the long-term one stores the last viewed ones.

- **Largest gap (LGap)**: In this technique, we first find the largest gap among the frequency values that

Threshold Technique	Fixed	LGap	Our technique	JFA
Average Precision	0.411	0.86	0.91	0.84

Table 3: Comparison of four techniques for discovering long-term interests.

are associated with the topics in the SBP, and then consider all the topics with frequency values larger than the largest gap as long-term interests.

- **Our technique**: This is in essence our proposed approach of modelling a dynamic user profile by using the frequency average and the standard deviation to discover long-term interests.
- **Just the frequency average (JFA)**: In this technique, we deploy our proposed approach of modelling a dynamic user profile to learn user interests by using just the average frequency as a threshold.

Unlike short-term interests that are discovered and learnt daily, long-term interests are discovered and learnt at the end of the experiment period (i.e. day 20). In Table 3, we show just the average precision of modelling long-term interests for all the user profiles for the four different threshold techniques. The results show that the Fixed technique recorded the lowest performance. This is because in (Grcar et al., 2005), all the last visited concepts are considered as long-term interests, which means that uninteresting and short-term interesting concepts are also treated as long-term interests. On the other hand, LGap surprisingly achieved a relatively good average precision of 0.86 which is better than both the Fixed and JFA. This result can be attributed to that most of the long-term interests in user profiles have higher frequency weights which are much higher than the short-term interests. However, one limitation of this technique which may not be obvious in our scenarios might appear when the list of concepts has more than one largest gap or the largest gap is between the uninteresting and short-term concepts, but not between short-term and long-term interests. In this case, the largest gap would likely capture inaccurate long-term interests from such a list. The results show that using both the frequency average and standard deviation instead of just the frequency average can provide better average precision, 0.91 and 0.84 respectively. This is because when using both frequency average and standard deviation just those long-term interests with the highest frequency weights are selected to be long-term interests, but not the short-term interests with high interest weights as in the case of using just the frequency average as a threshold.

Comparison Against Other State-of-the-Art Works. In this experiment, we compare the performance of our modelling system against two other approaches, namely: (Grcar et al., 2005) and (Li et al., 2007) (both were discussed in section 2). We test all the five scenarios which are described in more detail in appendix A. Figure 12 shows the comparison results between our work and these two approaches for all users over 20 days. It can be clearly seen that the approach of (Grcar et al., 2005) has achieved the lowest result with an average precision of 0.41 for all days. This is due to a number of limitations. The main limitation is that as *all* visited web pages are treated as interesting, even uninteresting ones, inevitably inaccurate interests would be captured and modelled. Another problem is that the user short-term profile is limited to a small number of interests, which means that when this gets full, many interesting concepts would be left out. This is further accentuated by the fact that abandoned interests are not removed from the profile.

The approach of (Li et al., 2007), performs slightly better with average precision over all days of 0.534. This result is due to different limitations. One problem is associated with the representation of a user profile where the interests are represented as concepts while the interest weights are the total number of visits. Even if the user had stopped visiting such interests as the number of visits would not change, the user interests

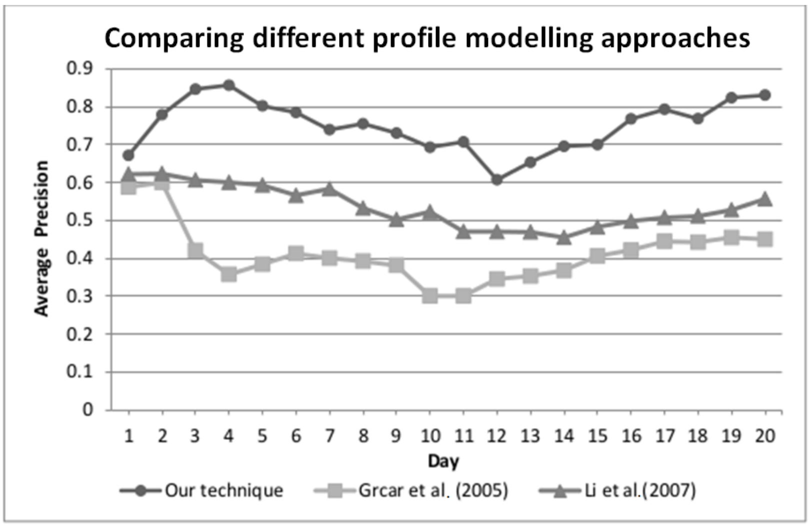


Figure 12: Comparison of three user profile modelling approaches.

would not be removed from the profiles. Another issue is that the size of the buffer in the short-term profiles is fixed, which leads to the same problem of capturing just a limited number of interests as in (Grcar et al., 2005). As a consequence, this approach could not model correct user profiles when a user is interested in a relatively big number of interests such as in day 7 in scenario 3. Our approach on the other hand, managed to achieve a good average precision of about 0.75 for all days. This result is due to that user profiles are not limited to a fixed number of interests, but the size adapts in the case of short-term profiles and is unlimited in long-term profiles. This performance can also be attributed to the ability to recognize the shifts and drifts in user interests as well as forgetting uninteresting concepts more effectively. Overall, although all the designed scenarios are different and exhibit behaviours that are highly changeable, our dynamic user profile managed to maintain good precision overall for all users during all days. This shows that our proposed methods are able to effectively capture user interests and adapt to different browsing behaviours.

6. Evaluation II: Dynamic User Profiling for Personalized Search

We wish to evaluate the performance of the learning and adaptation processes in the context of a personalization system. To this end, we implemented the proposed search personalization system in section 3.3 that utilizes the dynamic user profiles to provide re-ranked search results based on user interests.

We used the same reference ontology that we created for the first evaluation and used Google[7] as the search engine to retrieve the search results. As before, the Firefox browser with the Meetimer component was used to collect user behaviour. We invited 30 users to participate in an experiment over six days. All users were postgraduate computer science students, so they are expert computer users. They were asked to search and browse for interesting *Programming Languages* during the first three days. During the following three days, users are asked to search and browse for any interesting topic in the field of computer science. All users were asked to search and browse for at least three different interests during each of the three days. We do this so that we can examine the shift of interests and how our models adapt to this. For each day, users were asked to write down all the visited interests in order to compare them to the modelled dynamic

[7]http://www.google.com

# of queries submitted by all users	723
# of interests visited by all users	216
# of visited web pages by all users	1,290

Table 4: Collected data from all users.

user profiles. After six days, we collected all the 30 log files for all the 30 participants. Table 4 summarizes the collected information.

In the next step, we retrieved all the visited web pages for all users and extracted all their contents. All the web pages that belong to one interest were aggregated in one document. Then, in order to examine the personalization performance of our proposed search system, we asked each user to select three queries related to programming languages which were visited during the first three days, and to select another three queries related to the interests that were visited during the last three days of the experiment. For each of these queries, we retrieved the top 30 results from the Google search engine. We then randomized all of these results and asked users to select just the results that were relevant to each query. Users were free to click and navigate any result for more detail. Finally, we recorded the users' responses with regards to all of their queries.

6.1. Evaluating the Accuracy of Modelling User Profiles

We are interested in examining the accuracy of modelling user profiles and in particular, the quality of identifying and mapping visited web pages to concepts from the reference ontology. As users were asked to write down all the topics that they visited for each day, we compare these recorded topics to the concepts that were discovered in the dynamic users profiles using equation 13.

Figure 13 shows the average mapping accuracy results for all users in each day of the experiment for two approaches. The first approach is using our system with the GEW and 3C algorithms, while the second is using our system with the simple cosine similarity algorithm. According to Figure 13, the average accuracy for all users when our system uses the GEW and 3C algorithms is better than with the simple cosine similarity algorithm (0.791 and 0.602 respectively). In more detail, the accuracy of our model with the GEW and 3C on day 1 was only just 0.52. This could be attributed to that the dynamic user profiles have just started to learn and adapt to the users' browsing behaviour which means that they only have a small amount of information. However, the trend from day 2 to day 3 has improved significantly with accuracy being at 0.877 and 0.91 respectively. This is because after three days of learning, the profiles adapted effectively to the new interests which in turn improved the accuracy results. On day 4, unsurprisingly a sudden decrease occurred in the overall accuracy from 0.91 to 0.692. This drop was expected as users started to shift their browsing behaviour from navigating through programming languages to navigating through their own interests. Once users started to navigate more interests and browse more web pages on days 5 and 6, the overall accuracy experienced a considerable improvement from 0.692 on day 4 to 0.89 on day 6. This demonstrates that the sudden change of user interests would lower the performance of the dynamic user profile. Nevertheless, our system shows it can recover quickly as it effectively adapted to the abrupt changes. Finally, it is clear from this figure that using the GEW and 3C outperforms the cosine similarity on all days.

6.2. Evaluating the Re-ranking Approach

In this experiment, our goal is to evaluate the performance of the developed search personalization system that utilizes dynamic user profiles to re-rank search results. An effective re-ranking approach should

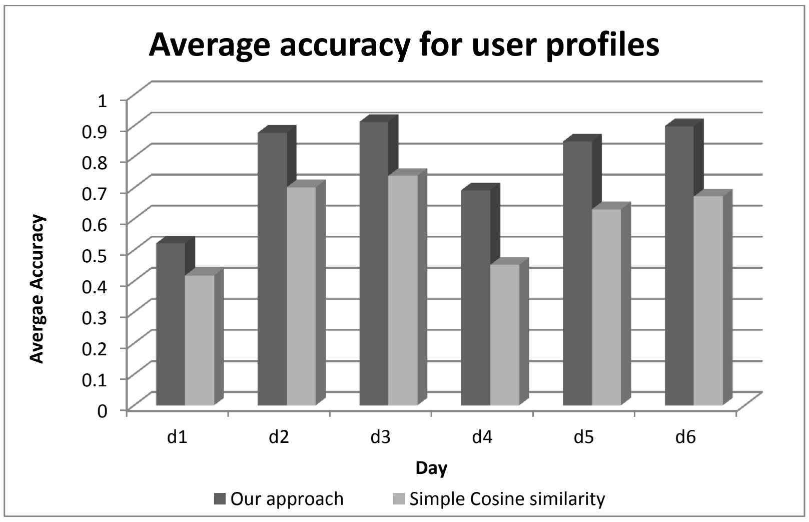


Figure 13: Average accuracy for each user profile over the six day period.

place the most relevant results at the top of the retrieved results. For this experiment, we evaluate the quality of our proposed system by comparing its performance against three different systems. The first system is the Google original ranking (GOR) that retrieves results for each query without any further processing. The second system is using our proposed search system (PSS) that utilizes dynamic user profiles. The third system computes the simple cosine similarity (SCS) between a user query and all the retrieved results from the Google search engine and then re-ranks these results based on their weights. The last system uses the rich concept descriptions in the reference ontology (RCD). We use a range of metrics to measure the quality of these re-ranking methods. The Average Ranking Error (ARE) is calculated as follows:

$$ARE(u_q) = \frac{\sum_{p \in R}^{n} p.position}{\mid Total\ number\ of\ p \mid} \tag{15}$$

Where u_q is a query that is sent by a user u, R is a set of all the retrieved results for the query u_q, $p.position$ is a position of a retrieved page p in the ranking list, and $\mid Total\, number\, of\, p \mid$ is the total number of results that are selected by users as related to their queries. This metric which has been used by many studies such as (Li et al., 2007; Dou et al., 2007) has one important advantage as it computes the error of the ranking order of any system in comparison to the actual ranking provided by users taking into account the order of all retrieved results. Therefore, in this metric, a smaller error indicates better performance. Figure 14 shows the ARE for all users for each system.

During the first three days, the PSS achieved the best overall performance in terms of ARE over all other three methods, while the RCD achieved the second best. The SCS and GOR, on the other hand, recorded lower results during the first three days. In more detail, the performance of our method (PSS) improved steadily from 13.53 on day 1 to 12.92 on day 3. This improvement is due to the fact that during the second and third days, our system managed to learn and collect sufficient information about the users' interests. The RCD, on the other hand, achieved lower performance than the PSS, but at the same time better performance than the SCS. This is because in this approach, the concepts in the reference ontology are associated with documents that represent these concepts. These documents contributed to the good performance as they hold rich information about the associated concepts which are used to re-rank the retrieved results. Finally,

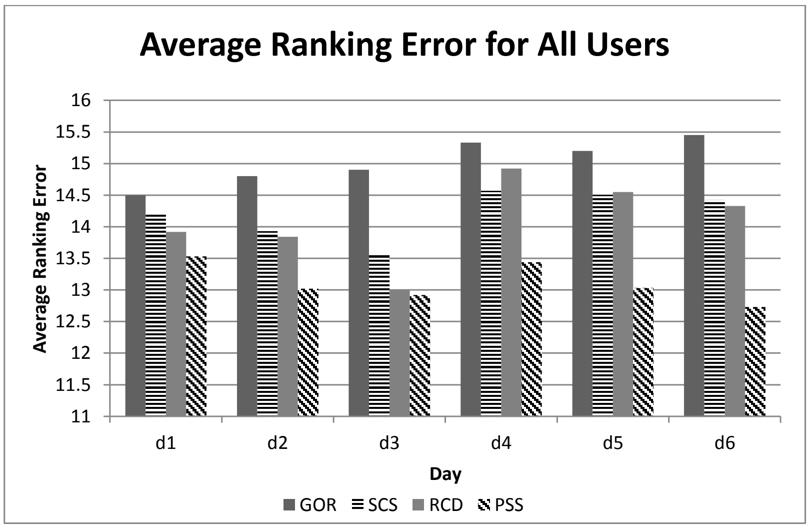


Figure 14: ARE across all four systems for all users over the six day period.

the GOR system has shown the worst performance and was the only approach that the performance kept getting worse during the second and third days. This is because during days 2 and 3, users started to navigate more interests, as well as visiting more web pages regarding the interests that they had visited earlier on day 1. As a result, new concepts were discovered, and more information about each concept was recorded. The emergence of new concepts resulted in the GOR mechanism scoring the lowest ranking average of about 14.8 and 14.9 on days 2 and 3 respectively, while all the other methods managed to improve their overall performance. On day 4, all four methods experienced a sudden drop in their performance. This trend was expected as on that day users started to shift their interests from navigating topics related to programming languages to topics related to their own interests in the field of computers. Similar to the results recorded on day 1, the PSS recorded better performance than the other methods and this performance kept improving during days 5 and 6 as a result of collecting more information about user interests.

Next, we look at the performance of all of the tested methods using the precision at N results ($P@N$) in order to examine the proportion of the relevant results ranked at different top N results. Unlike the previous metric, the order of the result is not important; what is important is to examine the users' satisfaction over the retrieved top N results for their submitted queries. We also use the Mean Average Precision (MAP) metric to report the mean of the ranking precision for all the queries submitted by all users. Figure 15 shows the precision performance of four methods at different top N results namely: P@5, P@10, P@15, P@20 and P@25, while Table 5 illustrates the MAP for these four methods.

The results in Figure 15 show that the PSS system achieved the best precision performance at all $P@N$ results. RCD and SCS scored the second and third best performances while GOR achieved the lowest precision. At P@5, the PSS method achieved the best precision result of 0.881 which represents about 12%, 13% and 18% improvement over the RCD, SCS and GOR respectively. This indicates the effectiveness of the system that provides a re-ranking search result based on the proposed method of modelling dynamic user profiles. We also note that the RCD achieved better performance than the other methods due to that all the retrieved results were re-ranked based on the information that is associated with the concepts in the reference ontology. Finally, both the SCS and GOR achieved the lowest precision performance at all $P@N$ results. When it comes to the MAP metric, the results reflect those of Figure 15. That is, the PSS has the highest MAP of 0.807, while the RCD and SCS scored the second and third best MAP (i.e. 0.748 and

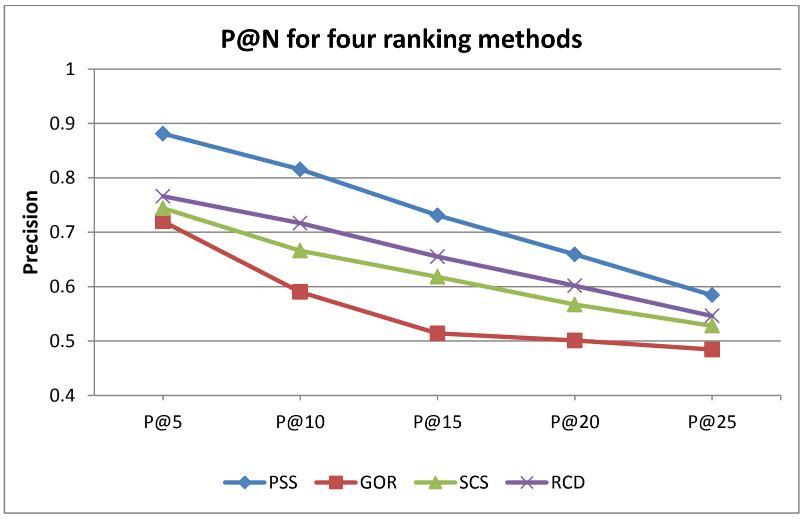


Figure 15: $P@N$ results for the four different ranking methods.

Method	MAP
GOR	0.677
PSS	0.807
SCS	0.73
RCD	0.748

Table 5: MAP results for the four ranking methods.

0.73 respectively), while GOR achieved the lowest MAP of 0.677. Overall, PSS effectively outperformed the three other methods, and was able to provide more accurate and reliable results when the user profiles had sufficient amounts of information. This demonstrates that the proposed dynamic user profile adapted effectively to the users' interests, and managed to provide an improved personalized search service.

7. Conclusions

In this paper, we presented a dynamic user profile modelling approach for web personalization. The starting point of our work was the identification of a number of issues in existing approaches, namely, the limitations of such systems in dealing with the user's changing interests over time, the lack of a clear delineation of the short and long-term aspects of user interests and weaknesses in modelling dynamic aspects of the user's behaviour. In our work, we aimed to address these issues. Firstly, the developed methods are able to deal with the constant change and transitions in a user's interests including sudden change. Secondly, we have developed methods that are able to capture to a satisfactory extent the differences in a user's short and long-term interests and the transition from one to another. These methods are also able to adapt based on the patterns of behaviour of individual users and hence we do not apply a *one-size-fits-all* approach. Finally, the methods developed have wider applicability and can be used in a number of domains and applications in which ongoing user behaviour can be captured and information about user preferences and interests can be extracted and modelled.

In more detail, our theoretical contribution is three-fold. Firstly, we introduce two mapping algorithms to improve the mapping process between web pages visited by the user that contain implicit information about his/her interests, and a reference ontology to explicitly represent these interests. Secondly, we introduce novel techniques to construct ontological short and long-term profiles that are tailored to the users, and adapt them based on their ongoing behaviour. Thirdly, within the methods for the construction and adaptation of user profiles, we introduce techniques to recognize and handle potential interest drift and interest shift in the user interests. Unlike other approaches in the literature such as (Challam et al., 2007; Zhang et al., 2007; Trajkova and Gauch, 2004), the methods developed are able to handle the dynamic nature of user interests and behaviours that change frequently. Our methods can handle the addition of new interests to a user profile, but also other more convoluted processes such as updating, gradually forgetting and deleting user interests. We also distinguish between long-term and short-term interests. Unlike approaches that suggest pre-defined thresholds to discover and store such interests (Grcar et al., 2005; Li et al., 2007), we presented mechanisms that are able to adapt to each user according to his/her behaviour. Another contribution is that the proposed methods are flexible and can be imported and used with diverse personalization systems. In this paper, we have illustrated this contribution by developing a re-ranking algorithm that uses our system for modelling dynamic user profiles to provide personalized search results to users.

In order to evaluate our work, we introduced a two-part evaluation. The first part aimed at evaluating the mapping and user profile modelling methods, while the second part aimed at evaluating the methods within a re-ranking personalization system. Both of these user-centered evaluations used the same reference ontology which was extracted from the ODP. The first evaluation showed that the methods developed are capable of mapping web pages to the reference ontology more accurately than other methods in the literature. In addition, we showed that by examining different user behaviours, the learning and adaptation strategies are capable of dynamically capturing user interests and forgetting the ones that are no longer of interest. We also demonstrated that our methods are able to deal with the interest-shift and interest-drift behaviours by minimizing the negative impact of these behaviours when modelling user profiles. In the second set of evaluations, we examined our work within a personalized search system. We found that our system achieved higher performance than the other search approaches. Our evaluation results demonstrated that the proposed methods can effectively capture user interests, adapt to the changes occurring in user behaviours and can enhance the performance of a personalization system.

Although in this work we have addressed some of the limitations of existing personalization methods and we have showed that our system is effective at learning and adapting dynamic user profiles, our proposed approach has a number of limitations. It is essential for our system to collect sufficient information about each user in order to learn user profiles, and without such information, the system would not be able to provide effective personalization services. This problem of having just a small amount of information is known as the *cold start* problem and is a typical problem in content-based personalization systems. In addition, in order for our adaptation methods to work effectively, the user needs to be using the system in an ongoing basis. If the user stops using the system for a period of time and returns while interests have changed significantly, the system will still be producing recommendations based on older information. As it is often the case with content-based systems, our system could suffer from the problem of over-specialisation. In essence, as the user profile adapts, the profile information becomes ever more specialised and the system may be unable to make novel suggestions to the user. Another issue is that the effectiveness of the techniques that we have proposed to infer and exploit semantic information from user profiles depend heavily on the richness and the quality of the reference ontology. We assumed the existence of such an ontology, but it is essential to note that the quality of the recommendations will depend on the quality and richness of the underlying reference ontology deployed. Another weakness of our work is that some of the models and mechanisms depend on parameters and settings that need to be pre-optimized. For example, in order to compute the semantic relatedness between ontological concepts, we need to pre-identify the weights of each relation type in an ontology. Similarly, the processes of learning, adapting and exploiting dynamic user profiles have a number of parameters that need to be identified. For our evaluation, such parameters were defined by running experiments using training datasets where the values and settings that provide the optimum results were selected. However, such a mechanism is too slow and requires conducting a large number of experiments. Hence when our methods are applied in different domains, these settings need to identified and carefully selected.

Ontological user profiles comprise a rich representation of user interests which can help us to understand the user needs in a more effective way and hence deliver better services. There are a number of avenues for future work and extensions that we would like to explore. Currently, although we are using ontologies to model the user profiles, we are not making use of the axioms that may be encoded into an ontology. Such axioms may allow more useful information to be extracted from the user profiles and more complex inferences to be made. We would like to explore how such axioms can play a role in further shaping the user profiles and also as part of the recommendation process.

With the huge growth of social networks, we believe that some of the developed methods can be extended and applied in order to provide social recommendations to users. In particular, we could extend the use of our methods to combine information included in social network profiles of individual users, but also make use of information in their connections' profiles in order to understand user preferences and interests better and also identify similarities with connections that can help enrich recommendations further. The latter could also help alleviate the problem of over-specialisation that our profiles may suffer from. In addition, such social network profiles contain multi-modal information which includes tags, videos, pictures, etc. and another direction that we could extend our work would be toward developing methods that could make use of different types of information.

In order to deal with the *cold start* issue, additional information can be used as suggested in (Middleton et al., 2004) which could be an external ontology that includes more information about users such as their job positions, publications, or the projects that they are working on. Such information could for instance be collated from sites like LinkedIn[8] (even if not available in an ontology format). Another potential solution would be to develop a hybrid recommendation technique which combines our ontological user profile with a collaborative filtering approach such as in (Basiri et al., 2010). It would be interesting to use such hybrid system to also take advantage of the features of both techniques in order to extend and enhance our system to provide more accurate and diverse services to users.

Acknowledgements

Dr Ahmad Hawalah was sponsored for his PhD studies by the University of Taibah, Medina, Kingdom of Saudi Arabia.

Anand, S.S., Kearney, P., Shapcott, M., 2007. Generating semantically enriched user profiles for web personalization. ACM Transactions on Internet Technology 7.

Baeza-Yates, R., Ribeiro-Neto, B., 2011. Modern Information Retrieval: The Concepts and Technology behind Search (2nd Edition). 2 ed., Addison-Wesley Professional.

Basiri, J., Shakery, A., Moshiri, B., Hayat, M., 2010. Alleviating the cold-start problem of recommender systems using a new hybrid approach, in: Proceedings of the 5th International Symposium on Telecommunications (IST 2010), pp. 962–967.

Blanco-Fernandez, Y., Nores, M.L., Gil-Solla, A., Cabrer, M.R., Arias, J.J.P., 2011. Exploring synergies between content-based filtering and spreading activation techniques in knowledge-based recommender systems. Information Sciences 181, 4823–4846.

[8]https://uk.linkedin.com

Borlund, P., 2003. The IIR evaluation model: a framework for evaluation of interactive information retrieval systems. Information Research 8.

Cantador, I., Fernandez, M., Vallet, D., Castells, P., Picault, J., Ribire, M., 2008. A multi-purpose ontology-based approach for personalised content filtering and retrieval, in: Advances in Semantic Media Adaptation and Personalization. Springer Verlag. volume 93, pp. 25—51. Volume title: Advances in Semantic Media Adaptation and Personalization.

Challam, V., Gauch, S., Chandramouli, A., 2007. Contextual search using ontology-based user profiles, in: Large Scale Semantic Access to Content (Text, Image, Video, and Sound) (RIAO '07), Pittsburgh, Pennsylvania. pp. 612—617.

Chen, C.C., Chen, M.C., Sun, Y., 2002. PVA: A Self-Adaptive personal view agent. Journal of Intelligent Information Systems 18, 173—194.

Claypool, M., Le, P., Wased, M., Brown, D., 2001. Implicit interest indicators, in: Proceedings of the 6th International Conference on Intelligent User Interfaces, ACM. pp. 33—40.

Daoud, M., Tamine-Lechani, L., Boughanem, M., 2008. Learning user interests for a session-based personalized search, in: Proceedings of the Second International Symposium on Information Interaction in Context, ACM, New York, NY, USA. pp. 57—64.

Dou, Z., Song, R., Wen, J., 2007. A large-scale evaluation and analysis of personalized search strategies, in: Proceedings of the 16th International Conference on the World Wide Web, ACM, New York, NY, USA. pp. 581—590.

Eirinaki, M., Mavroeidis, D., Tsatsaronis, G., Vazirgiannis, M., 2006. Introducing semantics in web personalization: The role of ontologies, in: Ackermann, M., Berendt, B., Grobelnik, M., Hotho, A., Mladeni, D., Semeraro, G., Spiliopoulou, M., Stumme, G., Svtek, V., Someren, M. (Eds.), Semantics, Web and Mining. Springer. volume 4289, pp. 147–162.

Felden, C., Linden, M., 2007. Ontology-Based user profiling, in: Abramowicz, W. (Ed.), Business Information Systems. Springer Berlin / Heidelberg. volume 4439 of *Lecture Notes in Computer Science*, pp. 314—327.

Gao, Q., Yan, J., Liu, M., 2008. A semantic approach to recommendation system based on user ontology and spreading activation model, in: Proceedings of the 2008 IFIP International Conference on Network and Parallel Computing, pp. 488–492.

Gauch, S., Speretta, M., Chandramouli, A., Micarelli, A., Brusilovsky, P., Kobsa, A., Nejdl, W., 2007. User profiles for personalized information access the adaptive web, in: The Adaptive Web. Springer Berlin / Heidelberg. volume 4321, pp. 54—89.

Godoy, D., Amandi, A., 2009. Interest drifts in user profiling: A Relevance-Based approach and analysis of scenarios. The Computer Journal 52, 771—788.

Gorgoglione, M., Palmisano, C., Tuzhilin, A., 2006. Personalization in context: Does context matter when building personalized customer models?, in: Proceedings of the IEEE International Conference on Data Mining, pp. 222—231.

Grcar, M., Mladeni, D., Grobelnik, M., 2005. User profiling for interest-focused browsing history, in: Proceedings of the Workshop on End User Aspects of the Semantic Web (in conjunction with the 2nd European Semantic Web Conference), pp. 99–109.

Hawalah, A., Fasli, M., 2011a. Improving the mapping process in ontology-based user profiles for web personalization systems, in: Proceedings of the International Conference or Agents and Artificial Intelligence ICAART, pp. 321—328.

Hawalah, A., Fasli, M., 2011b. A multi-agent system using ontological user profiles for dynamic user modelling, in: Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, pp. 430—437.

Huang, Z., Chung, W., Chen, H., 2004. A graph model for E-Commerce recommender systems. Journal of the American Society for Information Science and Technology 55, 259—274.

Kim, J., Kim, J., Kim, C., 2007. Ontology-Based user preference modeling for enhancing interoperability in personalized services, in: Stephanidis, C. (Ed.), Universal Access in Human-Computer Interaction. Applications and Services. Springer, pp. 903—912.

Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., Riedl, J., Volume, H., 1997. Grouplens: Applying collaborative filtering to usenet news. Communications of the ACM 40, 77—87.

Li, L., Yang, Z., Wang, B., Kitsuregawa, M., 2007. Dynamic adaptation strategies for long-term and short-term user profile to personalize search, in: Proceedings of the Joint 9th Asia-Pacific Web and 8th International Conference on Web-age Information Management Conference on Advances in Data and Web Management, pp. 228—240.

Liang, T.P., Yang, Y.F., Chen, D.N., Ku, Y.C., 2008. A semantic-expansion approach to personalized knowledge recommendation. Decision Support Systems 45, 401–412.

Liu, F., Yu, C., Meng, W., 2002. Personalized web search by mapping user queries to categories, in: Proceedings of the Eleventh International Conference on Information and Knowledge Management (CIKM), ACM, New York, NY, USA. pp. 558—565.

Liu, W., Jin, F., Zhang, X., 2008. Ontology-based user modeling for e-commerce system, in: Proceedings of the Third International Conference on Pervasive Computing and Applications (ICPCA) 2008, IEEE. pp. 260–263.

Middleton, S.E., Shadbolt, N.R., De Roure, D.C., 2004. Ontological user profiling in recommender systems. ACM Transactions on Information Systems (TOIS) 22, 54—88.

Montaner, M., 2001. A taxonomy of personalized agents on the internet. Technical Report, TR-2001-05-1, Departament d'Electronica, Informatica i Automatica. Universitat de .

Mooney, R.J., Bennett, P.N., Roy, L., 1998. Book recommending using text categorization with extracted information, in: Papers from 1998 Workshop on Recommender systems, AAAI Press. pp. 49–54.

Mooney, R.J., Roy, L., 2000. Content-based book recommending using learning for text categorization, in: Proceedings of the Fifth ACM Conference on Digital Libraries, ACM, New York, NY, USA. pp. 195–204.

MozillaWiki, 2010. User:Mconnor/Past/PlacesFrecency - MozillaWiki. See https://wiki.mozilla.org/User:Mconnor/PlacesFrecency.

Pan, X., Wang, Z., Gu, X., 2007. Context-Based adaptive personalized web search for improving information retrieval effectiveness, in: Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing, IEEE. pp. 5427—5430.

Picault, J., Ribiere, M., 2008. An empirical user profile adaptation mechanism that reacts to shifts of interests, in: Proceedings of the European Conference in Artificial Intelligence ECAI.

Pignotti, E., Edwards, P., Grimnes, G.A., 2004. Context aware personalised service delivery, in: Proceedings of the European Conference in Artificial Intelligence ECAI 2004, Valencia. pp. 1077–1078.

Porter, M.F., 1997. An algorithm for suffix stripping, in: Sparck Jones, K., Willett, P. (Eds.), Readings in Information Retrieval. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 313—316.

Razmerita, L., Lytras, M.D., 2008. Ontology-Based user modelling personalization: Analyzing the requirements of a semantic learning portal, in: Lytras, M.D., Carroll, J.M., Damiani, E., Tennyson, R.D. (Eds.), Emerging Technologies and Information Systems for the Knowledge Society. Springer. volume 5288, pp. 354—363.

Shani, G., Gunawardana, A., 2011. Evaluating recommendation systems, in: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (Eds.), Recommender Systems Handbook. Springer, pp. 257–297.

Sieg, A., Mobasher, B., Burke, R., 2007. Ontological user profiles for representing context in web search, in: 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology Workshops, IEEE. pp. 91—94.

Speretta, M., Gauch, S., 2005. Personalized search based on user search histories, in: Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, IEEE. pp. 622—628.

Sugiyama, K., Hatano, K., Yoshikawa, M., 2004. Adaptive web search based on user profile constructed without any effort from users, in: Proceedings of the 13th International Conference on the World Wide Web WWW '04, New York, NY, USA. p. 675.

Sun, J., Xie, Y., 2009. A recommender system based on web data mining for personalized E-Learning, in: Proceedings of the International Conference on Information Engineering and Computer Science, 2009. ICIECS, IEEE. pp. 1—4.

Trajkova, J., Gauch, S., 2004. Improving Ontology-Based user profiles, in: Proceedings of Large Scale Semantic Access to Content (Text, Image, Video, and Sound) RIAO, pp. 380—389.

Weng, S.S., Chang, H.L., 2008. Using ontology network analysis for research document recommendation. Expert Systems with Applications 34, 1857–1869.

Wu, K.L., Aggarwal, C.C., Yu, P.S., 2001. Personalization with dynamic profiler, in: Advanced Issues of E-Commerce and Web-Based Information Systems, Third International Workshop on WECWIS 2001, IEEE. pp. 12–20.

Xu, K., Zhu, M., Zhang, D., Gu, T., 2008. Context-aware content filtering & presentation for pervasive & mobile information systems, in: Proceedings of the 1st International Conference on Ambient Media and Systems, pp. 1–20.

Yang, Y., Padmanabhan, B., Rajagopalan, B., Deshmukh, A., 2005. Evaluation of online personalization systems: A survey of evaluation schemes and a knowledge-based approach. Journal of Electronic Commerce Research 6, 112–122.

Zhang, H., Song, Y., Song, H., 2007. Construction of Ontology-Based user model for web personalization, in: Proceedings of the User Modeling Conference, pp. 67—76.

Zhuhadar, L., Nasraoui, O., 2008. Personalized cluster-based semantically enriched web search for e-learning, in: Proceedings of the 2nd International Workshop on Ontologies and Information Systems for the Semantic Web, ACM, New York, NY, USA. pp. 105—112.

AppendixA. The Evaluation Scenarios

In this appendix, we present the five scenarios that were used as part of the first evaluation to address different user browsing behaviours.

Scenario (1): 'Normal behaviour': In the first scenario, we look at how our model would learn and adapt to normal behaviour that consists of uninteresting topics and short-term and long-term interests. Figure A.16 represents this scenario. This scenario consists of a total of 12 topics and 126 tasks (i.e. some of the tasks are repeated in different days). Users were asked to complete as many tasks as possible during 20 days. All the tasks in this scenario have been designed to simulate three browsing behaviours. The first behaviour is browsing uninteresting topics (e.g. topics 2, 4, 6, 8, 10 and 12 in table 1). The tasks in these topics occur in a short period and their occurrence is very low. In the second behaviour, we simulate a browsing behaviour when a user has long-term interests in some topics. In this scenario, topics 1 and 3 simulate this behaviour

Day / Topic	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Type
Topic 1	0	1	2	3	1	3	2	0	2	1	1	2	2	0	0	1	2	3	2	0	Long
Topic 2	2	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	2	0	Uninteresting
Topic 3	2	3	4	2	1	2	0	1	1	0	0	1	1	0	1	2	2	0	1	1	Long
Topic 4	0	0	0	0	2	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	Uninteresting
Topic 5	4	3	3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Short
Topic 6	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	Uninteresting
Topic 7	0	0	0	0	0	0	0	0	0	0	0	0	1	2	2	4	1	2	2	1	Short
Topic 8	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	Uninteresting
Topic 9	0	0	0	0	0	0	0	0	0	0	1	1	2	3	2	2	2	1	0	0	Short
Topic 10	0	0	0	1	0	0	0	0	0	0	0	0	2	0	0	0	0	0	2	0	Uninteresting
Topic 11	2	1	2	3	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	Short
Topic 12	0	2	0	0	0	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	Uninteresting

Figure A.16: Scenario (1): Normal Behaviour.

Day / Topic	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Type
Topic 1	0	2	3	1	1	1	1	0	0	0	0	0	1	2	3	4	3	2	2	1	Long
Topic 2	2	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	2	0	Uninteresting
Topic 3	3	3	4	4	3	2	1	0	0	0	0	0	1	2	2	2	2	3	2	2	Long
Topic 4	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	Uninteresting
Topic 5	2	2	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Short
Topic 6	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	Uninteresting
Topic 7	0	0	0	0	3	2	2	0	0	0	0	0	2	3	3	1	1	0	0	0	Short
Topic 8	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	Uninteresting
Topic 9	0	0	0	0	0	0	0	0	0	0	0	0	1	2	3	1	1	0	0	0	Short
Topic 10	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	2	0	Uninteresting
Topic 11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4	3	3	1	1	Short
Topic 12	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	Uninteresting

Figure A.17: Scenario (2): Browsing-stopped.

as both these topics occur over a long period and their occurrence is very high. In the final behaviour, we simulate a user behaviour when he has short-term interests in some topics. Examples of this behaviour are topics 5, 7, 9 and 11 in Figure A.16 which appear for a short period and their occurrence is high.
Scenario (2): 'Browsing-stopped behaviour': The second scenario again examines how our system would learn and adapt to normal behaviour. However, the difference in this scenario is that in the second third of the experiment's duration (i.e. from day 8 to day 12) users will not be asked to complete any task. The main aim of such behaviour is to see how our system reacts when users suddenly stop browsing the Internet for a short while and then return to their normal browsing behaviour (i.e. from day 13). Figure A.17 shows the 'browsing-stopped' behaviour scenario.
Scenario (3): 'Gradual interest-drift behaviour': Scenario (3) examines interest drift. For this purpose, we simulate a browsing behaviour where user interests are gradually changed from a set of topics to new topics. In order for the interest drift to be gradual, we designed the tasks in such a way so that short-term topics (i.e. topics 3, 5, 7, 9 and 12) start as interesting topics, and then they become less interesting after a while and just before a user starts to drift his interests to new topics. Figure A.18 shows the gradual interest drift scenario.

Scenario(4): 'Sudden interest-shift behaviour': Scenario (4) has been designed to test our system

Day / Topic	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Type
Topic 1	0	0	1	2	3	3	2	0	1	0	0	0	1	2	4	2	1	1	0	0	Long
Topic 2	2	0	0	0	0	0	0	1	0	0	0	0	2	0	0	0	0	0	2	0	Uninteresting
Topic 3	2	3	4	3	3	2	1	1	0	0	0	0	0	0	0	0	0	0	0	0	Short
Topic 4	0	0	0	0	2	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	Uninteresting
Topic 5	0	0	0	0	0	0	2	4	4	4	3	1	1	1	0	0	0	0	0	0	Short
Topic 6	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	Uninteresting
Topic 7	0	0	0	0	0	0	4	4	3	3	2	1	1	0	0	0	0	0	0	0	Short
Topic 8	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	Uninteresting
Topic 9	0	0	0	0	0	0	0	0	0	0	1	1	2	3	2	2	2	1	1	1	Short
Topic 10	0	0	0	1	0	0	0	0	0	0	0	0	2	0	0	0	0	0	2	0	Uninteresting
Topic 11	1	2	3	3	4	1	2	1	0	0	3	4	2	2	3	2	1	1	2	1	Long
Topic 12	0	1	1	2	3	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0	Short

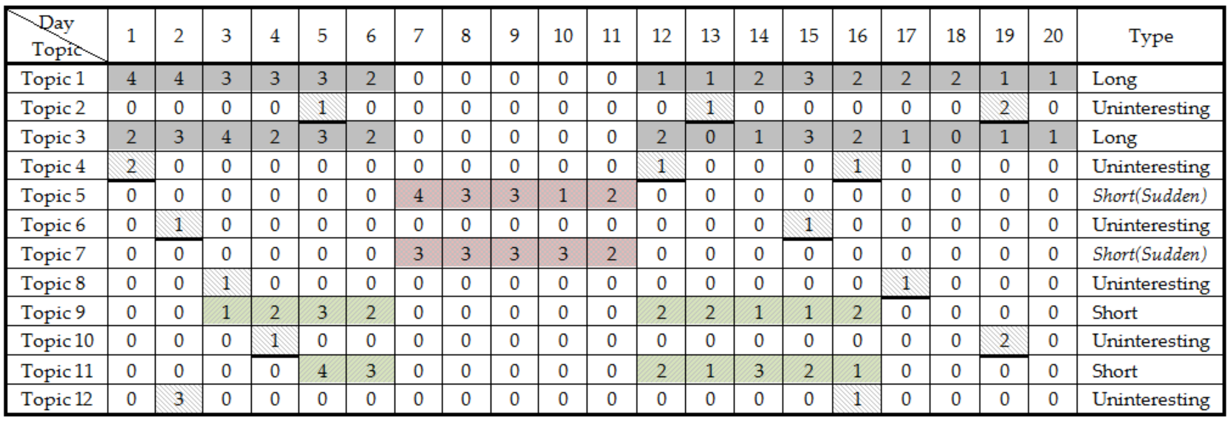
Figure A.18: Scenario (3): Gradual interest-drift.

Day / Topic	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Type
Topic 1	4	4	3	3	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Short
Topic 2	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	2	0	Uninteresting
Topic 3	2	3	4	2	3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Short
Topic 4	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	Uninteresting
Topic 5	0	0	0	0	0	0	4	3	3	1	2	0	0	0	0	0	0	0	0	0	Short
Topic 6	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	Uninteresting
Topic 7	0	0	0	0	0	0	3	3	3	3	2	0	0	0	0	0	0	0	0	0	Short
Topic 8	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	Uninteresting
Topic 9	0	0	0	0	0	0	0	0	0	0	0	2	2	1	3	2	1	1	1	1	Short
Topic 10	0	0	0	1	0	0	0	0	0	0	3	0	0	0	0	0	0	0	2	0	Uninteresting
Topic 11	0	0	0	0	0	0	0	0	0	0	0	4	3	3	3	3	2	2	2	1	Short
Topic 12	0	3	0	0	0	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	Uninteresting

Figure A.19: Scenario (4): Sudden interest-shift.

when user interests suddenly shift to new ones. In this scenario, the interest-shift occurs three times: from day 1 to day 6, from day 7 to day 11 and from day 12 to day 20 (see Figure A.19). In order to simulate the interest-shift behaviour, we designed all the tasks to be stopped suddenly, and new tasks about new topics take place abruptly. Our system would have to adapt to such behaviour in terms of recognizing new interests and forgetting uninteresting concepts rapidly.

Scenario (5): 'Interrupted browsing behaviour': In scenario (5), we simulate a situation when a user's regular behaviour is interrupted. In addition, a user might have regular interests on some topics, but in some situations he might lose this interest temporarily as he might get interested in new topics. But after a while, the user would get back to his regular behaviour and get interested again in the previous topics. For example, a user might have a long-term interest in reading topics related to his work, but if he decided to take a holiday in France for example, his interests would probably shift to topics related to France. However, once he gets back from his holiday, he would most likely return to his previous behaviour by browsing again for the earlier topics that related to his work. Figure A.20 shows this scenario.



Day / Topic	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Type
Topic 1	4	4	3	3	3	2	0	0	0	0	0	1	1	2	3	2	2	2	1	1	Long
Topic 2	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	2	0	Uninteresting
Topic 3	2	3	4	2	3	2	0	0	0	0	0	2	0	1	3	2	1	0	1	1	Long
Topic 4	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	Uninteresting
Topic 5	0	0	0	0	0	0	4	3	3	1	2	0	0	0	0	0	0	0	0	0	*Short(Sudden)*
Topic 6	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	Uninteresting
Topic 7	0	0	0	0	0	0	3	3	3	3	2	0	0	0	0	0	0	0	0	0	*Short(Sudden)*
Topic 8	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	Uninteresting
Topic 9	0	0	1	2	3	2	0	0	0	0	0	2	2	1	1	2	0	0	0	0	Short
Topic 10	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	Uninteresting
Topic 11	0	0	0	0	4	3	0	0	0	0	0	2	1	3	2	1	0	0	0	0	Short
Topic 12	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	Uninteresting

Figure A.20: Scenario (5): Interrupted browsing behaviour.

AppendixB. Example of adaptation of ontological user profile

In this appendix, we present a brief example of how a user's ontological profile adapts based on the interaction of the user with the system and through browsing topics. This example has been taken from a real user profile that was created during our experiment as described in section 5. The profile shown in Figures B.21-B.23 represents only a fraction of the actual user profile and is provided for illustrative purposes. The numbers indicated alongside the concepts are the interest weights.

Figure B.21 illustrates a fragment of the user's profile after this has been mapped to the reference ontology ODP on Day 1 of the experiment. The user has the following short-term interests: HCI, Dell, CSS, and Data mining. The average accuracy of the user profile on that day was 0.66 and this was due to the limited information present in the user's profile.

In Figure B.22, we can see the adapted fragment of the user profile on Day 5 of the experiment. Our approach captured the users interests as being in: HCI, Dell, CSS and XML. By Day 5 certain concepts that were ascertained as being of no interest to the user such as Data Mining and Database were forgotten as they just appeared only once during Day 1. The concept Directories was also forgotten as the user showed more interest on the HCI concept. The accuracy during day 5 was in fact close to 1 for this specific user.

Finally, Figure B.23 shows the adapted user profile by Day 15 of the experiment. By Day 15, the user profile has experienced a number of changes:

- As the user developed new short-term interests (AI and AJAX) this caused the accuracy to drop. The logfiles reveal that there was much noise in the collected user interests which caused the *Insert Agent* to add wrong concepts to the user profile (i.e. Javascript, Agent and Fuzzy).
- Long-term interests were recognized as XML and HCI as they recorded high frequency weights (.793 and .778 respectively) compared to the short-term interests.
- The *Forget Agent* processed and forgot three concepts that used to be short-term interests (i.e. Data Mining, Dell and CSS).
- Two concepts were deleted from the user profile as the user did not show any interest in them. These concepts are Database and Directories and their frecency weights were less than the calculated threshold for concepts to remain in the user profile (even as forgotten concepts).

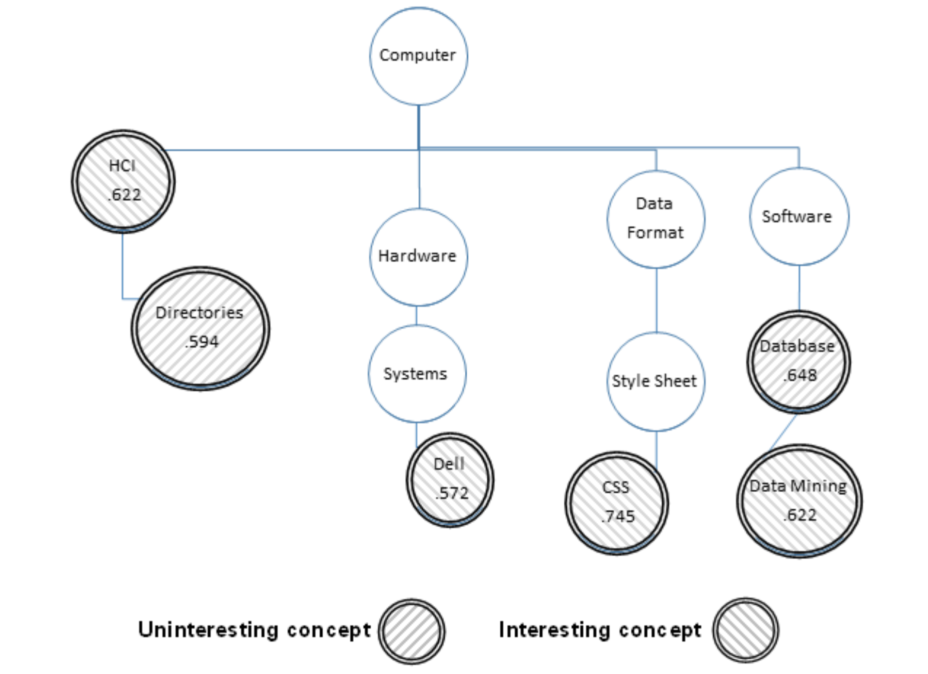


Figure B.21: Day 1: Ontological user profile after mapping user interests to reference ontology.

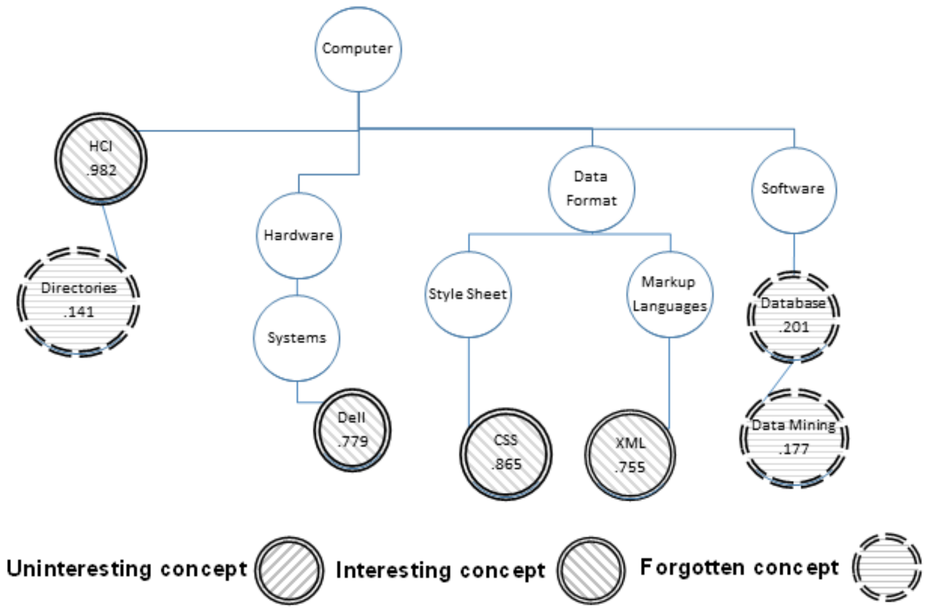


Figure B.22: Day 5: Adapted ontological user profile after interaction with the system.

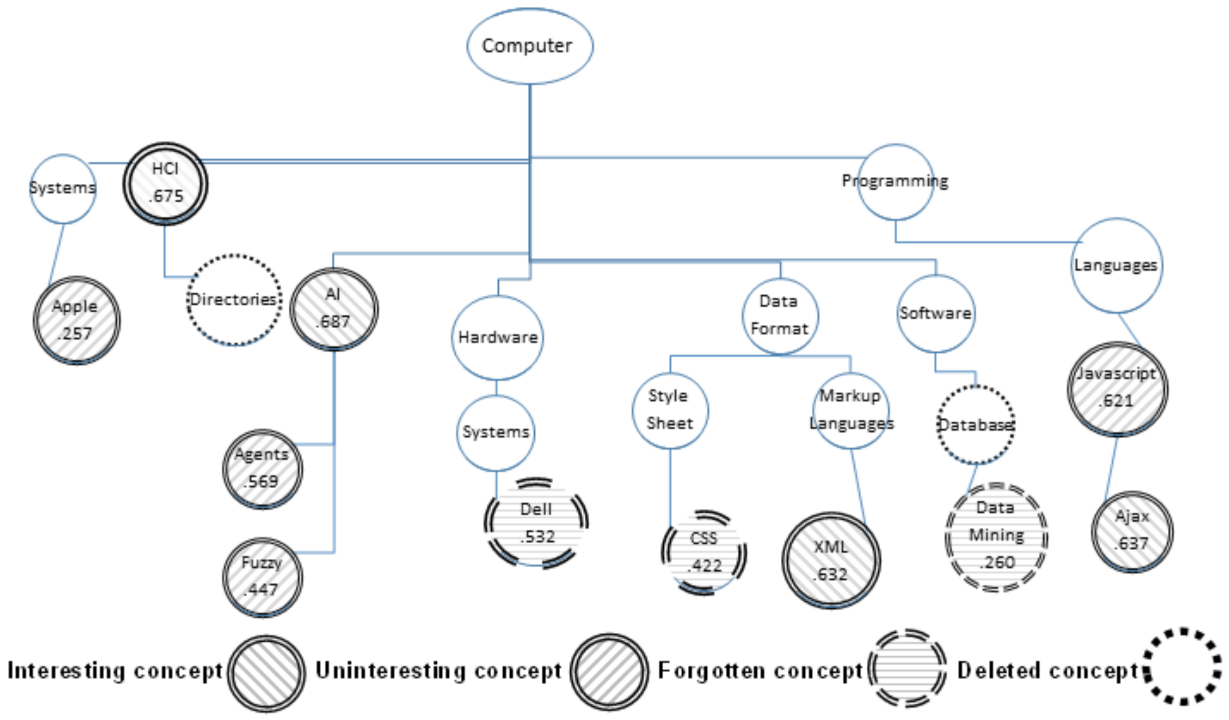


Figure B.23: Day 15: Adapted ontological user profile after further interaction with the system.