A Semi-supervised Topic-based User Model for Web Information Visualization

Shibli Saleheen **Wei Lai**

Faculty of Science, Engineering and Technology
Swinburne University of Technology
Hawthorn, Victoria, Australia
Email: {ssaleheen,wlai}@swin.edu.au

Abstract

Web graph is a very effective tool to visualize web information. To serve user specific visualization and to reduce the size, personalization is applied to web graphs, by integrating user interests in filtering, graph generation and clustering. Modelling of the user interests is the key aspect to achieve personalization. Keyword-based user models are being used frequently for faster results. However, shortcomings of keyword-based models include lack of semantics that leads to inefficient similarity measurement and categorization. An ontological user profile can solve the polysemy problem inherent in a keyword-based profile. Because ontological profiles are mostly domain specific, they cannot be used efficiently in general web space which has information from diverse domains. This paper presents a topic-based hierarchical user model to address cross-domain interest-based visualization. The development and update procedures of the model consult the WordNet and/or the user. In connection with that, user interest-based measures are provided for graph generation, such as term and document similarity as well as document relatedness. These play an important role in user interest-based filtering and clustering. An experiment shows the effectiveness of the model over its keyword-based counterpart.

Keywords: Semi-supervised User Model; Topic-based User Model; Personalization; Web Data Visualization

1 Introduction

The World Wide Web(WWW) plays a vital role in people's daily lives. Users browse and navigate the web space to meet the need for information. Nevertheless, the continuous growth in size and diversity of information have made it difficult for end users to mine interested information from the WWW. Visualization of the web information via graph which unfurls the inherent relationships is useful for end users. However, reducing the size of the web graph by filtering unwanted information and grouping similar ones together is very important from the end user viewpoint. Exiting filtering (Huang & Lai 2006) and clustering (Gao & Lai 2008) applied to web information networks do not include user interests in during calculation and hence end up presenting the same web graph to all users. Because end users demonstrate diversity in their needs, these approaches lack the expected effectiveness, which calls for integration of user interests in generating web graphs.

User modelling is an integral part of personalized services. Although personalization via user modelling has started long back in the late 1970s (Kobsa 2007) before the introduction of the WWW, recently it has become a core component of many web applications (Bilenko & Richardson 2011). Researchers have established various methodologies for constructing user models and applying them to achieve personalization. Because of unique and diverse requirement of systems, user modelling is generally considered as a domain specific task. Being a key component for personalization, a careful construct of the user model is necessary to achieve the highest possible outcome for end users in filtering, clustering and representation of information. The remaining challenges include choosing the best fitted user model which performs effectively to satisfy users' real time needs. Construction of user models to tailor a web graph for individual's needs is not an exception. Moreover, being semi-structured in nature, it is difficult to apply personalization techniques to web information. As a result, an effective user model is yet to be devised for multi domain information sources like the WWW.

Our previous work (Saleheen & Lai 2014) presents a user interest-based web graph to accommodate the needs of the user during visualization. An architecture to construct and visualize such a web graph is presented by another work (Saleheen & Lai 2013) of us. The user model is applied to the graph generation process, which filters out irrelevant web documents and generates relationships among the rest of the documents; to produce the initial graph; and to the clustering process to group similar documents[refer to the paper (Saleheen & Lai 2013)]. An extension of the architecture is presented in Figure 1. The keyword-based approach successfully reflects the user interests on web graphs, however exposes the polysemy problem as described in (Gauch et al. 2007) because it does not account the keyword relationships. To overcome the above stated challenges, this paper presents a user model capitalizing on the WordNet (Miller 1995) to relate interests of the user. Options are provided to the user to step up and make the necessary changes to the interest profile. The topic-based model efficiently works with cross-domain web data and does the filtering and graph generation more effectively.

The roadmap of the paper is as follows: a brief discussion on the existing user modelling schemes and the motivation for a different user model are described in Section 2; the skeletal representation of the user model is outlined in Section 3 as well as the initial development of the user profiles; Section 4 details the user feedback analysis; Section 5 presents the method-

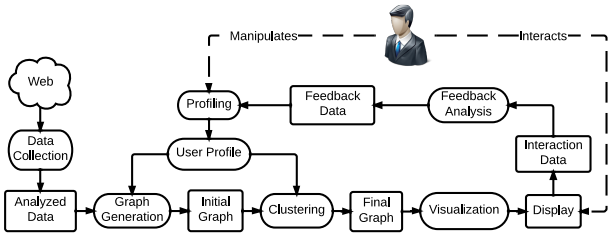

Figure 1: Architecture of Interest-based Visualization

ology to update the user profile based on the received user feedback; the user interest-based similarity measures to relate web documents are described in Section 6; the cold-start problem is addressed in Section 7; an experiment comparing the user model to its keyword-based counterpart is presented in Section 8 and the paper concludes by stating the benefits and contributions and future work in Section 9.

2 Background and Motivation

To the best of our knowledge, personalization is yet to be applied to the context of web graphs. However, user modelling is not rare, it is found in many other domains. The works that are related to user modelling in various domains are described instead. As described in (Joung et al. 2009), contexts of information that contribute in user modelling are divided into four classes: General Information, Events, Preferences and Social Network. The general information class captures personal details such as name, contact and demographic of the user. The event class stores user activities and status information. The preferences class aggregates the user interests, whereas the social network class explains the user's connections and communications with others. Among the four classes, the preference class is the most influential to achieve personalization. The preferences or interests are usually extracted from various sources such as browsed documents (Bakalov et al. 2009) and socially tagged information (Michlmayr & Cayzer 2007).

Generally, user profiles are represented by three different formats (Gauch et al. 2007) for personalized services: keyword, semantic network and concept-based representations. Among these, the most common is the keyword-based representation which may contain some slots, but generally includes pairs of keyword and weight where the keyword is regarded as the interest and the weight as the corresponding degree of that interest for a user. The keywords are generally extracted from the web or provided by the user and weights are computed by measures like TF-IDF. To address the polysemy problem inherent in keyword based profiles, semantic network representation of keywords is used. In such representations, a node is regarded as a concept and two concepts are connected to each other based on their co-occurrences, i.e., the model represents the connections among nodes as semantic links which include co-occurring words of a document and the degree of affinity between a topic and a document. A concept-based profile exhibits representation similar to semantic network-based profile, but differs from the former in node representations, i.e., considers nodes as abstract topics rather than specific keywords. The concepts are normally arranged in a hierarchy where a top level concept presents more abstraction than a lower level concept. However, a concept-based user profile can be easily converted to a set of vectors of weighted keywords which resembles keyword-based profile.

Apart from modelling an individual user, the personalization can also be approached for groups of users. The work of Li et al. (Li et al. 2011) is an example of such approaches where a system is developed to recommend personalized news. Concurrent issues of user modelling include gaining effectiveness in cross-domain systems and dealing with the privacy of the user data. An attempt to overcome cross-domain adaptability challenge is found in the work of Orlandi et al. (Orlandi et al. 2012) where they collect information about the user from the social web to construct different user profiles for the same user. Later, these profiles are aggregated to build a generalized user profile. Bilenko and Richardson (Bilenko & Richardson 2011) propose a keyword-based client side user profile to subside the issues regarding privacy and user control that are frequently raised by the server side counterparts. In addition to these, Ghosh and Dekhil (Ghosh & Dekhil 2009) point out the cold-start problem and describe it as 'Profile Sparseness', which often occurs during the first usage of a system by a user. They also find that typical user profiles suffer from 'User Profile Personae'. In other words, the contexts of the user and the environment that influence the personalization process are very difficult to overcome. They demonstrate that the introduction of a 'Profile Manager' can reduce the degree of effect produced by the above stated challenges.

Ontology is widely used nowadays to reduce semantic gaps that reside in keyword-based user profiles. In addition to the world knowledge-base, Tao et al. (Tao et al. 2011) emphasise the need of integrating the local knowledge repository to the construction of ontology-based user profiles. They use Library of Congress Subject Headings(LCSH) to infer the ontology which is later used to derive the relationships among concepts. They classify user profiles into three categories: interviewing, semi-interviewing and non-interviewing. The near perfect user profiles are constructed from the direct user involvement, i.e., the interviewing profiles. On the other hand, Ahn et al. (Ahn et al. 2007) demonstrate that user's ability to control user profile information can be harmful to system performance, if it is not used with caution. To bring balance between these two, a call for the need of semi-interviewing user profiles is placed.

Kim and Chan (Kim & Chan 2008) present an interest-based hierarchical user profile which is obtained by grouping similar interests extracted from the visited documents. The implicit profile development process follows agglomerative approach where a high-level interest presents large term-list and eventually the root of the hierarchy contains all interests of the domain. In other words, a high-level concept is actually representing more interest-terms than a low-level one. The model does not account the semantic super/sub relationships among interest-terms. The user models that are described above have their respective shortcomings which include: 1) most of them are built in unsupervised fashion which means updates on profiles are conducted without consulting any renowned knowledge-base; 2) the effectiveness of performance varies in different domains when a specific knowledge-base is used; and 3) some other models are too heavy to be used in real time processes.

The WWW is very dynamic in nature as new information is added frequently to it. This makes the polysemy problem more severe and creates difficulties for domain specific user models to adapt new information without enhancing the knowledge-base manually. As a consequence, the user model should aim to address the above stated challenges in interest-based visualization by carefully modelling the user. This work

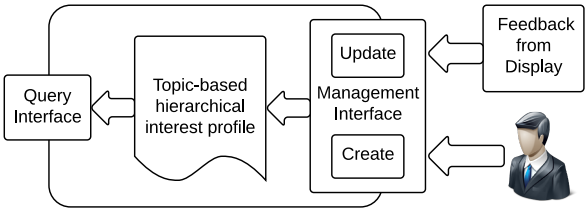

Figure 2: Architecture of User Modelling

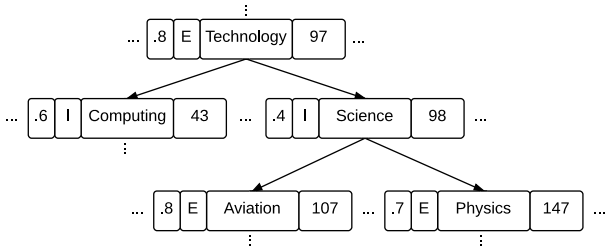

Figure 3: Excerpt of Topic-based User Profile

attempts to address the following key points during the development of the user model:

- The interest-based visualization is a real time system which involves user's direct interaction. Therefore, the model should be able to compute the necessary measurements as quick as possible. That is, the model should be reasonably fast while negotiating with the accuracy.
- For a new user, where no prior user profile is constructed, the model should be able to build the user profile from the scratch.
- Because this model is to visualize the web information which is heterogeneous in nature, the model should be able to work with different information domains.
- The user model should be able to keep itself updated automatically utilizing the feedback information produced during the navigation.
- The model should keep the sensitive user information such as browsing history, accessed URLs and their content undisclosed to public.

This approach is similar to the approach of Kim and Chan (Kim & Chan 2008), however, WordNet consultancy is incorporated to generate the topic hierarchy and to utilize semantic hypernym/hyponym relationships among the terms. To achieve interest-based visualization, two domain independent measures are devised to calculate term and document similarities applying the greedy approach. In addition to that, a measure to calculate document relatedness is also developed.

3 User Modelling

3.1 The Topic-based Model

To accommodate differences over users and serve the visualization according to the individual's needs, it is essential to model the user related information. The architecture presented in Figure 1 contains 'Profiling' and 'Feedback Analysis' modules which play important roles in user modelling.

Figure 2 presents the high level architecture of the user modelling scheme. A topic based hierarchical user profile resides at the centre of the architecture. Unlike keyword-based user profiles, the topic-based user profile unleashes a hierarchical representation of user interests where interest-topics are regarded as nodes and pose super/sub relationships to each other. The model also contains two interfaces to provide access to the hierarchy for other components and the user. The first one is the 'Management' interface which provides options to create and update, whereas the second one is the 'Query' interface which provides options to query the model for the measurements.

An excerpt of the topic-based user profile used in the system is shown in Figure 3. This excerpt shows that topics are organized in hierarchical fashion where 'Technology' is a super-topic of 'Computing', 'Science', etc. The topic 'Science' also has some sub-topics. Each node of the user profile is comprised of four parts. Apart from the topic-name, there is a weight score(ranges from 0 to 1) indicating the degree of interest in this topic. Another part is the mode-of-entry indicating the method of adding the topic. Types 'E' and 'I' correspond to explicit addition by the end user and implicit or calculated addition respectively. Finally, the frequency stores the number of times the topic has been visited so far. The weight score of a topic gets updated as the user uses the system, regardless of its mode of entry.

3.2 Development of Topic Hierarchy

The first task of the 'Management Interface' is to create or develop the user profile. It actuates when a new instance of the system is launched or a new user starts using the system. Generally, the development commences with the direct user input for some predefined categories. After that, gradually the hierarchy of the topic-based profile is created and weight scores for all topics are assigned. An interest of the user is added as a topic to the user profile either explicitly by the user or implicitly by the system.

3.2.1 Direct User Input

A user poses numerous amount of information for a system. The volume of user information is directly proportional to the accuracy of the personalization. Nevertheless, higher volume of information implies larger costs for both computation and time complexities. Therefore, a good modelling scheme should carefully select the relevant information about the user. Each of the four categories of user context described by Joung et al. (Joung et al. 2009) has its significance in different context-aware services. However, this model emphasizes on the preferences of a user because interest-based visualization mainly relies on the interests of the user. To get explicit user topics, it is necessary to get direct user input. First and foremost, the user answers few questions related to general information such as name, age, profession, location, etc. After that the user is also asked to enter interests and corresponding weight values. The first preference is automatically added as a sub-topic of the 'root'. All other subsequent topics are handled differently, which is described in the next section.

3.2.2 Implicit Development using WordNet

Regardless of the mode of entry, for adding the second and subsequent topics, it is needed to figure out the appropriate place in the hierarchy. The position lookup for a new topic can be accomplished by direct user input, i.e., the user is asked where to add the new topic. However, finding an appropriate position becomes challenging for a user as the size of the hierarchy grows. Therefore, it is beneficial for

Algorithm 1: Potential Positions for a Topic

```
   Input  : C_n, T(C, E)
   Output: P(C, d, sim)
1  begin
2      compute depth of C_n from WordNet root → depth(C_n)
3      set P(C, d, sim) ← Φ
4      forall the C ∈ T do
5          set sim ← WordNet similarity of C, C_n
6          if sim ≥ τ then
7              list common parents of C and C_n → CP
8              if C, C_n ∈ CP then
9                  set P ← S ∪ (C, exact, sim)
10             else if C ∈ CP then
11                 set P ← S ∪ (C, down^+, sim)
12             else if C_n ∈ CP then
13                 set P ← S ∪ (C, up^+, sim)
14             else
15                 compute depth of C from WordNet
                   root → depth(C)
16                 if depth(C_n) > depth(C) then
17                     set P ← S ∪ (C, down, sim)
18                 else if depth(C_n) < depth(C) then
19                     set P ← S ∪ (C, up, sim)
20                 else
21                     set P ← S ∪ (super(C), down, sim)
```

the end user to get suggestions about where to add. The model calculates potential positions to add a new topic as depicted in Algorithm 1:

- For each existing topic C of the hierarchy, the similarity score with the new topic C_n is calculated. The topics having similarity scores greater than the threshold τ are considered.
- A sub/super relationship using the common parents CP of topics C and C_n is determined. If the new topic C_n appears in the parent list, then C_n acts as a super topic of C in the hierarchy and vice versa.
- If none of the C or C_n is in the common parents' list CP, then the decision is taken whether C_n appears after/before C. According to the WordNet, the position is an indication of after/before positions is calculated. For example, the topic which has a lower depth from the 'root' in the WordNet comes before other topics.

Usually, Algorithm 1 generates multiple potential positions for the new topic C_n in the hierarchy as C_n can be similar to more than one topic. However, positions those fall in the same path of the hierarchy are replaced with the most relevant position. A potential position is represented as the tuple (C, d, sim), where C is an existing topic, d is the direction where C_n to be added and sim indicates the similarity score between C_n and C. There are six possible positions for C_n. The notations for d are: *exact* indicates C_n and C are same to some extent; up^+ and $down^+$ for C_n to C indicate direct super and sub topic relation respectively for C_n; and *up* represents indirect super-topic and *down* represents indirect sub-topic relationships with respect to C and $super(C)$ respectively. When C_n appears to be a sibling of C, it is expressed as a *down* of the super-topic of C, i.e., $super(C)$. Handling the *exact* direction just appends C_n with C and does not require further processing. Therefore, Algorithm 1 generates two sets of positions($\{up^+, up\}$) and ($\{down^+, down\}$) which should be processed further: 1) the set containing direct super/sub relationships and 2) the set where only indications of before/after are present. To deal with these, two different strategies are developed for computing suggestions.

Algorithm 2: Path Suggestions for a Topic

```
   Input  : P(C, d, sim)
   Output: S(p)
1  begin
2      set S(p) ← Φ
3      forall the C ∈ (P|d = up^+) in incremental depth(C)
       of the hierarchy do
4          remove all C_p from P where C_p is a member of the
           subtree with root = C
5          generate path p = root → super(C_p)
6          set S ← S ∪ p
7          remove C from P
8      forall the C ∈ (P|d = down^+) in decremental
       depth(C) of the hierarchy do
9          remove all C_c from P where C_c is a member of the
           path root → super(C)
10         generate path p = root → C
11         set S ← S ∪ p
12         remove C from P
13     repeat Lines 3 to 7 for d = up
14     repeat Lines 8 to 12 for d = down
```

Algorithm 3: Add a Topic to a Path

```
   Input  : T(C, E), C_n, p
   Output: T(C, E)
1  begin
2      set C ← C ∪ C_n, E ← E ∪ {e(end(p), C_n)}
3      list all topics C_s → N where super(C_s) = super(C_n)
4      forall the C_s ∈ N do
5          set E ← (E − {e(super(C_n), C_s)}) ∪ {e(C_n, C_s)}
```

Algorithm 2 describes the generation of paths for the potential positions. The list is processed in four phases, each for a specific direction. First the list C with up^+ is sorted according to depths of C from low to high. Theoretically, a super topic of C is also a super topic of any sub-topic of C. Therefore, for each topic C, all topics that appear in the list and are sub-topics of C are removed. C is also removed from P after the path $p = root \rightarrow super(C)$ is added to the suggested list $S(p)$. The second phase of the Algorithm 2 operates for $down^+$ directions. The list of all C of P, where C_n is a sub topic of C, is sorted by the depth of C from high to low. Again, a sub topic of C is also a sub topic of any super topic of C. With this, all super topics of C from the path *root* to C from P are removed as the new topic C_n is to be added after C.

In similar fashion, Algorithm 2 then executes the list of topics C which have *up* or *down* relationships with C_n. That is, both the topics C and C_n are similar to some extent, but do not pose any super/sub relationship in WordNet. However, the end user may want to establish a relationship between them as the user model is very tiny in comparison to the size of WordNet. A value l(typically less than 10 because it is easier to pick up from a short list) as the maximum size of the suggestion set is set.

The generated suggestion path list is presented to the end user to choose the paths to be added. The user may decide to add the topic in multiple positions if wishes. On top of that, the user may add new positions to be added for a topic. After the choices are made, the topic is added into the hierarchy using Algorithm 3. In this context, a path is always directed and starts from the root. According to Algorithm 3, the new topic, C_n, is added at the end of a path. However, few links may need to be adjusted to keep the hierarchy consistent. Any sibling C_s that appears as a direct sub-topic of C_n is repositioned. If the super-topic of C_n is denoted as $super(C_n)$, the link between $super(C_n)$ and C_s is removed and a link be-

tween C_n and C_s is established, i.e., C_n is considered as $super(C_s)$. After adding the topic, the weight score is taken from the user and inserted into the model.

4 Feedback Analysis

To provide relevant information to the end user in the course of time, it is very important to infer the need of the user effectively. Feedback analysis is a very important task to know the latest trends of the user and hence non-parallel to other features to keep the user profile updated. In the context of user interest-based visualization, the user feedback information is divided into two categories: implicit and explicit.

4.1 Implicit User Feedback

Implicit profile update depends on the implicit user feedback which is generated during the usage of the system. Depending on user activities of a session, the feedback analyser collects the feedback data to produce a suitable format containing potential operations for the profiler to the user profile. Implicit updates can affect the user profile by applying two types of operation: add and modify. In both cases, a path and a corresponding weight score are produced and sent to the 'management interface' of the 'profiling' component. However, before sending the data, the feedback analysis component analyses the gathered information and computes the relative weight scores for topics to be added or amended in the profile.

4.1.1 Browsing Feedback

During a session, a user visits a bunch of web pages which is the key source to collect feedback information as their content reflects the intention of the user. Following a user's visit of a page, two important factors are considered: (1) the duration of the visit, i.e., the dwelling time and (2) the page length, which is eventually substituted by the number of topics. Intuitively, if a user spends a long time on a page, the level of user interest of the terms of that page is increased. On the other hand, if a page is long, the influence of the time factor is decreased since the increase in time is likely to be the effect of the volume of information presented, not for the degree of interest (Gauch et al. 2003). Rather than considering the total dwelling time of a page, time spent per topic is important in this context to update the interested topics and their respective weight scores. Let us consider a web page has total N topics and T is the total dwelling time in seconds for the page. Therefore, the time spent for a topic is determined by the $timelengthfactor = log(T/N)$ (Gauch et al. 2003). It is natural for a user not be restricted in visiting only one page in a certain session. Moreover, a single topic can appear in multiple documents. To address these, it is necessary to calculate the effective weight of a topic t that appears in m documents. The formula to calculate the effective weight t is:

$$E_t = \frac{1}{m}\sum_{i=1}^{m} w_i \cdot timelengthfactor_i \qquad (1)$$

4.1.2 Exploration Feedback

During the journey of user interest-based visualization, a user not only browses interested web pages, but also generates various actions by interacting with the user interface. One of the important actions is exploring a cluster. The user explores a cluster only when the interest level of the topics of the cluster is high enough to attract the user. Because the user can return to another node or collapse the expanded cluster without visiting its child nodes, this feedback data does not contribute to generate new interest topics. The exploration feedback analysis calculates the potential increment of the interest level of a specific interest. For a topic t appearing in the cluster C, the incremental difference, δ is defined as:

$$\delta = \frac{n}{N} \cdot \frac{m}{M} \cdot \frac{n}{f} = \frac{mn^2}{fMN} \qquad (2)$$

where, n is the total number of appearances of t in C; m is the number of documents where t appears; N is the number of all topics in C; M is the total number of documents; and f is the current frequency of t.

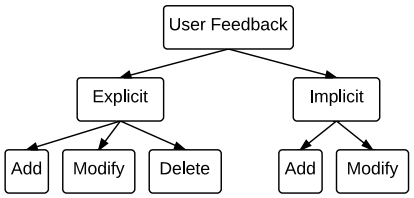

Figure 4: Feedback Types and Associated Operations

4.2 Explicit User Feedback

Explicit user feedback occurs when a user manually updates the user profile. An interface is provided to the user to access and edit the user profile. The user checks the concurrent interests of the profile hierarchy, and if misleading interests(in terms of topic and weight) are found as the effect of the implicit update, the user manually modifies the topics and/or their corresponding weight scores.

5 Updating The Model based on Feedback

The second task of the 'Management Interface' is to keep the user profile updated. Updating occurs during the use of the system and is based on either explicit or implicit user feedback. The update operation on a topic is divided into three categories: addition, modification and deletion. All three operations are available for the explicit user feedback or manual user input, however, for the implicit user feedback, the deletion of a topic is not performed. In case of the implicit update operation, weight scores of surrounding topics are usually affected(update on neighbour weight scores is described in Section 5.2). Figure 4 presents the types of user feedback and their associated operations in this context of user modelling.

5.1 Addition

In case a new topic is generated from the implicit feedback analysis, the user profile is updated by implicitly adding the topic to the hierarchy. The path selection process described in Section 3.2.2 is followed to construct the potential paths. However, for each document that contains the topic, only the path which has the maximum similarity score for the new topic is considered for addition. The maximum similarity score is calculated by comparing all topics of the document to all topics of the path. Because the implicit addition can occur only as the outcome of the browsing feedback, it is easy to get the other topics of the

document. If the topics of the path p and document d are denoted by the sets C_p and C_d respectively, the similarity of p and d is given as:

$$sim(p,d) = \begin{cases} \frac{\sum_{i=1}^{|C_p|}\sum_{j=1}^{|C_d|} sim(C_{pi},C_{dj})}{|C_p||C_d|} & \text{multi-topic} \\ \frac{\sum_{i=1}^{|C_p|} sim(C_{pi},C_d)}{|C_p|} & \text{single topic} \end{cases} \quad (3)$$

The explicit addition of a topic in the profile is manually performed by the user who may choose to use the path suggestion technique or insert the topic directly. The user is requested to enter the interest and its weight score.

5.2 Modification

Modification of the user profile refers to updating the weight score of an interest. Similar to the addition, modification occurs from both explicit and implicit user feedbacks. However, unlike the implicit addition which occurs only for the browsing feedback, implicit modification takes place as results of both the browsing and the exploration feedbacks.

The E_t of Equation 1 is used to calculate the final weight score for a potential topic in the case of the browsing feedback. Therefore, if the topic t has a weight score w and a frequency f, the final weight score w_f is calculated for t as:

$$w_f = \begin{cases} \frac{E_t + w*f}{f+m} & \text{browsing feedback} \\ w+\delta & \text{exploration feedback} \end{cases} \quad (4)$$

The w_f score achieved from the Equation 4 is used to update the weight score of a topic. However, when a topic of the hierarchy is updated, surrounding topics are also affected. For example, if the weight score of the topic 'Science' is updated, the weight score of 'Technology' should be amended because both of them are related to each other. Therefore, it is important to decide how an update on a weight score is propagated to the neighbouring topics.

Because a lower level topic of the hierarchy is more specific than a higher level topic, the effect of an update to the subordinate topics of a modified topic is not significant. Updating the weight scores of the super-topics is sufficient (Cena et al. 2011). In this model, the path that is selected for modification, lists the higher level topics of the modified interest. Therefore, updating the scores of the topics of the selected path is sufficient to reflect an update adequately. The propagation of score is calculated using the difference of the actual and the modified scores as $\Delta = (w_f - w)$. An updated score for a topic at η levels higher than the modified one is calculated as:

$$w_u = \frac{w\Delta}{w_m \eta^2 (n_s + 1)} \quad (5)$$

where, w is the current score of the super-topic, w_m is the actual score of the modified topic and n_s is the number of siblings of the modified topic. From the Equation 5 it is clear that, the updated score of a super-topic is inversely proportional to eta and n_s, which implies that the effect of modification is reduced as the propagation goes further levels or the number of siblings of the topic is higher.

5.3 Deletion

Deletion in the user model is only reserved for the explicit use by the end user. If the end user wants to delete one of the listed interests, it is instantly deleted. However, the links of the subordinate nodes are re-arranged by adding the subordinate nodes to the super topic of the deleted topic.

6 User Interest-based Measures

Because user interest-based visualization requires the similarity of nodes to be measured with respect to the needs of the individual, it is efficient to accomplish the similarity computations in the user model. This model provides three similarity measures for terms and documents, as parts of the 'Query Interface', which are described below.

6.1 Term Similarity

Term similarity is the measure to figure out the topic of the user profile that is the most relevant to a document term and their degree of similarity.

Definition 1 *User Interest-based Term Similarity (UITS): A real number which indicates the degree of similarity of a document term to the user interest profile. If T is the set of topics of a document and P is the user profile, F be a function that assigns a real number ranging from 0 to 1 to every topic $t \in T$. That is, $F : T \rightarrow [0, 1]$. The $UITS$ score of topic t is denoted by $F(t)$ where $0 \leq F(t) \leq 1$.*

Measuring the term similarity is not straightforward as a topic can appear in more than one path. For example, the topic 'Jaguar' can appear as a sub-topic of 'Animal' and 'Car'. Again, a term can be matched with several topics of the profile. Therefore, it is important to determine the topic that thematically matches best with the document term. Typically, the extracted terms represent the thematic meaning of a document. Using this phenomenon, the term similarity is measured.

First, it is needed to determine the candidate topics for a given term by checking all paths of the profile. According to Bilenko et al. (Bilenko & Richardson 2011), optimal profile construction is NP-hard and can be efficiently solved by a greedy approach to find out a near optimal solution. Similarly, checking for all paths with regards to a document is certainly of high computational complexity. For this, a greedy approach to select the candidate topics is invoked. Before applying the greedy approach, it is important to determine the characteristics of the model:

- The root of the profile has an interest score of 0.
- A sub-topic in the profile is more specific than its parent, that is the immediate super-topic.
- If a path of the profile is relevant to a document, all the possible sub-paths are also relevant to that document.

The given term is matched to the topics of the interest profile according to the bottom-up approach. That is, a lower level topic is matched first and the similarity value is stored. Algorithm 4 describes the process of selecting of candidate paths.

Once a level is finished matching, the term is matched with the parents of all candidate topics. A sub-topic is replaced by the super-topic if the later poses higher similarity score than the former. The process continues until all matched topics get greater similarity value than their corresponding super-topics. The paths to root from the super-topics of candidates(which have a similarity score greater than threshold γ) are computed and stored.

Algorithm 4: Selection of Candidate Paths

```
   Input  : T(C, E), t
   Output: P
1  begin
2      compute l ← depth(T)
3      set Candidates(candidate, score), P ← Φ
4      while l ≠ 0 or ∃c ∈ C|sim(c, t) < sim(super(c), t) do
5          forall the c ∈ C of T at level l do
6              compute sim_c ← sim(c, t)
7              set Candidates ← Candidates ∪ {c, sim_c}
8          set l ← l − 1
9      forall the
       candidate ∈ (Candidates|sim(candidate, t) > γ) do
10         compute path p as candidate → root
11         set P ← P ∪ p
```

Algorithm 5: Best Path Selection

```
   Input  : P, d, t
   Output: path_b(c → root), score
1  begin
2      sort P by length from low to high
3      forall the p ∈ P do
4          set P ← P − {p} if ∃q ∈ P where p is sub-path of q
5      forall the p ∈ P do
6          set p_temp ← ({p − c_f} → root) where c_f is the
           first topic of p
7          compute sim(p_temp, d − {t}) by Equation 3
8      set path_b ← arg max_{p_temp} sim(p_temp, d − {t})|p ∈ P
9      set score ← sim(c_f(path_b), t)
```

Algorithm 6: Document Similarity

```
   Input  : T(C, E), d
   Output: BestPaths({t, P}), score
1  begin
2      compute l ← depth(T)
3      set BestPaths, TempPaths({t, P}), d_accessed ← Φ
4      while l > 0 and d ≠ Φ do
5          set C_temp ← all c ∈ C of T at level l
6          forall the c ∈ C_temp do
7              compute sim_c^m = w_c · arg max_t sim(c, t)|t ∈ d
8              if sim_c^m > ψ then
9                  add path p = c → root to the path-list P
                   of t in TempPaths
10                 set d_accessed ← d_accessed ∪ {t}
11         set d ← d − d_accessed
12     forall the t ∈ TempPaths do
13         set BestPaths ← BestPaths ∪ p where p is the
           best path for t obtained by Algorithm 5
14     set score ← average of the scores of t ∈ BestPaths
```

Algorithm 7: User Interested Path List

```
   Input  : d, T(C, E)
   Output: P
1  begin
2      set P ← Φ
3      forall the t ∈ d do
4          set p ← best path of T using Algorithm 4 and 5
5          change scores of all elements of p by Equation 6
6          set P ← P ∪ {p}
```

Algorithm 5 computes the best matched path from the list. Paths that are sub-paths of a path with a higher score are removed from the list as the last characteristic of the model holds. Equation 3 is used to compute the similarity between a candidate path and the document topic-set except t. The path which has the maximum similarity score is chosen as final.

6.2 Document Similarity

Document similarity measures the relatedness of a document with the interest profile. This measure computes the best matched user interests for the document and their corresponding degree of relatedness.

Definition 2 *User Interest-based Document Similarity (UIDS): A real number which indicates the degree of interest of the user for a document. Let D is the set of documents; S be a function that assigns a real number ranging from 0 to 1 to every document $d \in D$. That is, $S : D \rightarrow [0, 1]$. The $UIDS$ of a document d is denoted by $S(d)$ where $0 \leq S(d) \leq 1$.*

It is common for a document to contain thematically different terms and usually a document-term is matched with more than one topic of the interest profile. As a consequence, computing the relatedness of a document must consider all candidate paths for different terms to select the best ones. This becomes very costly considering time and computational complexities. The candidate path selection process needs to be selective to reduce the complexities.

Algorithm 6 describes the process to compute document similarity by selecting the possible best paths for each term in a greedy fashion. Because the characteristics described in Section 6.1 hold, the topics of a document are matched with the interests, according to the bottom-up approach to find out the most relevant terms. Paths for these terms are constructed and stored against the respective terms. Because of the second characteristic of the model, if a document term is matched at the lower level of the hierarchy it is not considered for higher level. All potential paths are added to the corresponding path-set of a term and the best path for each term is computed using Algorithm 5. The arithmetic average of the maximum similarity scores of all most relevant terms is regarded as the final relevancy score for the document.

6.3 Document Relatedness

Document relatedness is used to determine the relationship between two documents with respect to the interest profile. Later this measure is used to decide the existence of edges among nodes of the web graph.

Definition 3 *User Interest-based Document Relatedness (UIDR): A real number to represent the degree of similarity between two documents d_i and d_j of the document-set considering the user focused interests at a specific time. Let R be a function that assigns a real number ranging from 0 to 1 to every pair of documents d_i, d_j in D then it is written $R : |D| \times |D| \rightarrow [0, 1]$.*

To calculate relatedness between documents with respect to the interest profile, the respective hierarchies are matched. To construct the document hierarchy as per the profile, related paths for a document are extracted. The path-list extraction of a document, d according to the hierarchy, T is accomplished as described in Algorithm 7:

- For each term in the document, the best matched path of the hierarchy is extracted by using Algorithms 4 and 5 consecutively.(Line 4).
- Because a document term has a weight score, which indicates the strength of the term in the document and in the corpus, the scores of all elements of the chosen path are changed using the following equation:

$$w_e^{updated} = w_e \cdot w_t \tag{6}$$

where w_e is interest score of the path element and w_t is weight score of the topic in the document.

Algorithm 8: Construct Document Hierarchy

```
   Input  : P
   Output: H
1  begin
2      set H ← Φ
3      forall the p ∈ P do
4          set p_rest ← p − root(p)
5          set current path element, e_c ← child(root(P))
6          set hierarchy element, e_h ← root(H)
7          set current hierarchy children, N_c ← children(e_h)
8          while N_c ∩ {e_c} ≠ Φ do
9              set e_c ← child(e_c), e_h ← e_c,
                 N_c ← children(e_h), root(p_rest) ← e_c
10         add p_rest, to H under e_h
11     forall the element e ∈ H do
12         update weight score of e using Equation 7
```

Algorithm 9: Relatedness between Hierarchies

```
   Input  : H_1, H_2
   Output: score, p_best
1  begin
2      set P_L(path, score) ← Φ
3      forall the p ∈ H_1 do
4          find longest matched path p_m in H_2
5          set P_L ← P_L ∪ {p_m}
6      update all element scores of p_m ∈ P_L using Equation 8
7      set p_best ← the P_L path with the max leaf node score
8      set score ← score(p_best)
```

The hierarchy for each document is constructed from the extracted path list. The Algorithm 8 details the process of hierarchy construction as follows:

- The best position to insert a path p of the path list P to the hierarchy is determined(Lines 5 - 9). Following the top-down approach, the process checks for a match between path-topic and hierarchy topics. The process continues iteratively starting from root until a non-match is found. During each iteration, the path and matched hierarchy topics are updated. The path topic is replaced by its child, while the hierarchy topic by the matched child of its children.
- After finding the best position, the remaining rest of the path, p_{rest} is directly added to the hierarchy under the last matched item, e_h.
- The weight scores of all nodes of the hierarchy are updated after the completion of the construction. The weight score is stored for each match between a hierarchy topic and a path topic during the construction process. The numerical average of these weights is considered as the final weight score of the document hierarchy topic:

$$w_{final} = \frac{1}{n} \cdot \sum w_{mtc} \tag{7}$$

  where n is the total number of matches of this topic during hierarchy construction and w_{mtc} is the weight score of the topic in each match.

Using the document hierarchies of two documents, the $UIDR$ is calculated. The idea is to find out the overlapped portion of document hierarchies and take the best path from the overlapped hierarchy. The best path in this context is the path with the maximum leaf node weight score. The detailed steps are described in Algorithm 9 as follows:

- Each element of a hierarchy is matched with all elements of the other to find the best match. In other words, each path p of H_1 is matched with all paths of H_2. The longest match is considered as the best match in this context. All the best matched paths are listed for further processing.
- For all best matched paths p_m of P_L, the elements of p_m get updated scores. The updated weight score of an element is the numerical average of the weight scores taken from the hierarchies for that element. If e is an element of p_m and the weight scores of e in H_1 and H_2 are $w(e_{H_1})$ and $w(e_{H_2})$ respectively, the updated score of e is:

$$w(e_{updated}) = \frac{w(e_{H_1}) + w(e_{H_2})}{2} \tag{8}$$

- The best path p_{best} on the list P_L is selected as final path based on the higher leaf node score. The score of the leaf node is the relatedness score of the documents and the path itself is the measure how they(documents/hierarchies) are related.

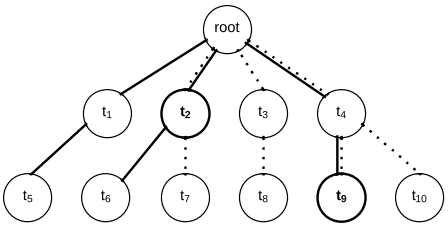

Figure 5: Illustration of Hierarchical Similarity

Figure 5 presents an illustration of the relatedness calculation between two hierarchies. The first hierarchy , i.e., for document d_1 is presented by solid edges and the second, i.e., for d_2 is by dotted edges. The topic sets for document d_1 and d_2 are $\{t_5, t_6, t_9\}$ and $\{t_7, t_8, t_9, t_{10}\}$ respectively. Two hierarchies are overlapped to find out longest path matches. For this example, the longest matched paths are $t_2 \rightarrow root$ and $t_9 \rightarrow t_4 \rightarrow root$. The best matched(highest scoring comparing the scores of t_2 and t_9) path of these two is selected as final path.

7 Handling the Cold-Start Problem

In order to produce user interest-based web graph, it is important to have a user profile constructed prior to graph generation. However, the system can encounter a situation where it lacks an interest profile. For instance, during the first use of the system the user can forward without creating a user profile manually. In such case, the system has no choice except running without the user interest information because according to this model, the relatedness between documents is calculated using interest-based hierarchies.

The entire process of measuring the relatedness of two documents is replaced in this special circumstance. The computation is accomplished using the content information of the documents. The implicit hierarchy generation process described in Section 3.2.2 is invoked to generate the document hierarchy using document-terms instead of interests. Later, the hierarchies are passed to Algorithm 9 to calculate the relatedness. The document hierarchy generation process consists the following steps:

- For each term t of document d, the topic position is calculated by the Algorithm 1.
- Path suggestions for a term t are computed by using Algorithm 2.
- The term t is added to the hierarchy using Algorithm 3. However, unlike Section 3.2.2, the user is not asked to choose the positions to add them.

Table 1: Filtering in Different Methods

Approach	Information Type	Profile after # of visits				
		2	4	6	8	10
Topic	Total Filtered	28	23	20	18	18
	False Positives	19	15	10	7	7
	False Negatives	4	5	3	2	2
Keyword	Total Filtered	25	21	16	12	9
	False Positives	15	10	6	5	4
	False Negatives	3	2	3	6	8

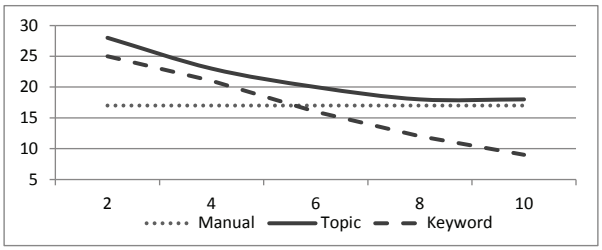

Figure 6: Number of Filtered Documents

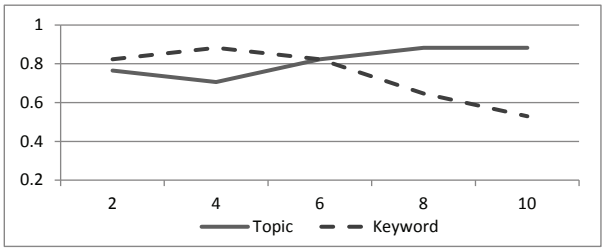

Figure 7: Filtering accuracy

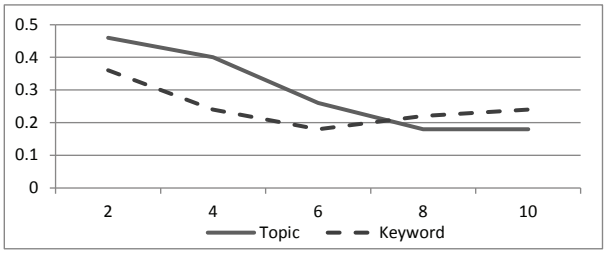

Figure 8: Filtering Error

Table 2: Document Relatedness in Different Methods

Approach	Information Type	Profile after # of visits				
		2	4	6	8	10
Topic	Correctly Established	10	13	17	23	25
	Erroneous Links	25	23	17	8	3
Keyword	Correctly Established	9	11	18	20	21
	Erroneous Links	26	25	20	16	12

8 Evaluation

8.1 Experiment

This experiment simulates the user model for the implicit user feedback. The OpenCalais[1] is used to extract the terms of a web-document that are used to populate the interest profile and to compare web documents against the profile. From the OpenCalais API, an extracted term comes with a relevancy score, which is considered as the weight score for the term. The web page navigation is simulated by employing a random function to generate the page number and the duration of the visit(between 60 to 120 seconds).

It is assumed that the user is interested in the 'technology' domain and no user profile is constructed prior to the first use of the system. A total of 15 technology related pages are downloaded from the BBC news site's Technology[2] section for using in the development phase of the interest profile. The random function selects one of the 15 pages each time it is invoked. The user is assumed to visit the website for 10 times. For each visit, the interest profile is updated using the extracted terms. However, the user model is stored only after every 2 visits for comparison.

To test the user model, another website, i.e., the CNN[3] news site is considered. In the second phase of the experiment, a total of 50 web pages are downloaded in 5 categories: 'Tech', 'Business', 'Travel', 'World Sport' and 'Entertainment'(10 from each category). The set of web documents is filtered according to the $UIDS$ scores. After that, the $UIDR$ calculation is applied to the remaining documents of the set to establish relationships among them. For the both cases the threshold is set to 0.3.

8.2 Results and Discussion

In Section 8.1 the web documents are filtered and linked successfully which proves the usability of the topic-based user model. However, to compare the effectiveness, it is needed to have another reference model operated with the same experimental setup. The keyword based model of our previous work (Saleheen & Lai 2013) is chosen. Filtering documents and establishing relationships among the rest of them are also performed manually for comparison. The number of filtered documents in the manual process is 17. Table 1 presents the statistics of document filtering for different approaches and different interest profiles.

Figure 6 presents the comparison of the number of filtered documents for different interest profiles between the keyword and topic-based approaches. The dotted line shows the number of manually filtered documents which is fixed for the entire experiment.

The set of filtered documents includes some false positives(those should not be, but filtered) and false negatives(those should be filtered, but are not) when compared to the manual approach. Based on false negatives, the accuracy of filtering is measured. The comparison of accuracy is presented in Figure 7, which shows that the accuracy gets significantly better for topic-based approach as the user navigates.

The comparison of errors in filtering is presented in Figure 8. Both the false positive and false negative counts contribute in the error calculation. It is evident from the figure that both approaches have the tendency of diminishing error in filtering with the navigation. However, in later stages, the error-rate in keyword-based approach gets higher due to the increase in false negatives.

To establish relationships among the left over documents after filtering, the $UIDR$ is used. The manual approach establishes 31 relationships. Table 2 presents correctly and erroneously established links for both the topic and the keyword-based approaches.

The accuracy of an approach is measured by comparing the correctly identified links to the manually annotated links. Figure 9 compares the accuracy of the topic and keyword-based approaches for interest profiles constructed after different number of visits.

Similar to filtering, both false positives and negatives contribute to the error counting in establishing relationships. Figure 10 compares the error-rate of both approaches. It is noticeable from the figure that

[1]Open Calais, http://www.opencalais.com/
[2]BBC News Technology, http://www.bbc.com/news/technology/
[3]CNN, http://edition.cnn.com/

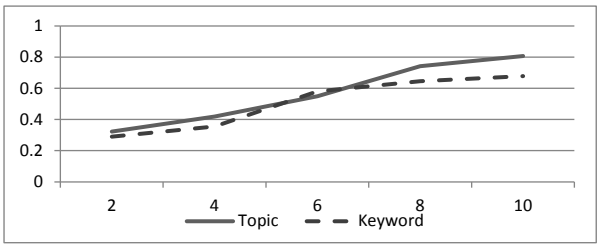

Figure 9: Comparison of Successful Detection

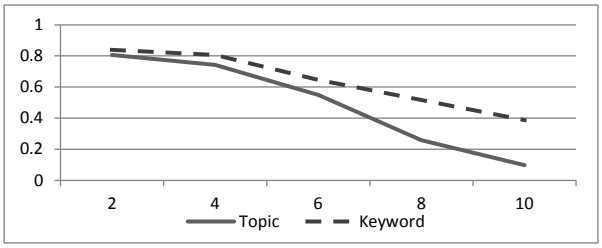

Figure 10: Error Comparison in Relatedness

keyword-based approach is more prone to error than the topic-based approach, though both of them show diminishing rate in error as the user navigates.

In a nutshell, the experiment reflects that the topic-based approach performs better than the keyword-based one in filtering and establishing relationships among web documents. The topic-based approach gets better as the number of visits increases.

9 Conclusion and Future Work

Despite of numerous attempts to establish user models for personalization services, a standard model is yet to appear in information sources such as web due to the dynamic and heterogeneous nature. As a result, each personalization service attempts to define its own model according to its specific needs. Existing user models are either too heavy to be included in a real time system or mostly specific to an information domain. Accumulating the nature of the source information and the browsing patterns of the user, a fast and cross-domain model is needed to be developed.

In this paper, a topic-based hierarchical interest profile for user interest-based web information visualization is presented. The schemes for developing and updating interest profiles are provided which encompasses both automatic and manual processes. The model consults with the WordNet to construct the hierarchy of interests and to keep it updated during the automatic manoeuvre. The model also provides classification of user interactions and respective operations for the collected feedback data. Faster methods to measure term and document relevancy are provided by the adaptation of the greedy approach to enhance the usability of the model. The model works on the client side to negotiate the privacy issues raised by server side models. It is also designed to operate as a standalone system, i.e., without the user involvement. The experimental results prove the effectiveness of the model in user interest-based visualization.

There are several directions to extend this work. Extensive experiment will be conducted in various information domains and usability test will be performed involving the real users. Because the extracted terms play an important role in the accuracy of personalization, the topic extraction will be analysed accounting the interests of the profile. A cross-domain ontology will be developed to construct the interest hierarchy in a more robust way.

References

Ahn, J.-w., Brusilovsky, P., Grady, J., He, D. & Syn, S. Y. (2007), Open user profiles for adaptive news systems: Help or harm?, WWW, pp. 11–20.

Bakalov, F., König-Ries, B., Nauerz, A. & Welsch, M. (2009), A hybrid approach to identifying user interests in web portals, *in* 'IICS', pp. 123–134.

Bilenko, M. & Richardson, M. (2011), Predictive client-side profiles for personalized advertising, KDD, pp. 413–421.

Cena, F., Likavec, S. & Osborne, F. (2011), Propagating user interests in ontology-based user model, *in* 'AI* IA', pp. 299–311.

Gao, J. & Lai, W. (2008), Visualizing blogsphere using content based clusters, *in* 'WI-IAT', pp. 832–835.

Gauch, S., Chaffee, J. & Pretschner, A. (2003), 'Ontology-based personalized search and browsing', *WIAS* **1**(3-4), 219–234.

Gauch, S., Speretta, M., Chandramouli, A. & Micarelli, A. (2007), User profiles for personalized information access, *in* P. Brusilovsky, A. Kobsa & W. Nejdl, eds, 'The Adaptive Web', pp. 54–89.

Ghosh, R. & Dekhil, M. (2009), Discovering user profiles, *in* 'WWW', pp. 1233–1234.

Huang, X. & Lai, W. (2006), 'Clustering graphs for visualization via node similarities', *Vis. Lang. and Comp.* **17**(3), 225–253.

Joung, Y., El Zarki, M. & Jain, R. (2009), A user model for personalization services, *in* 'ICDIM', pp. 1–6.

Kim, H.-R. & Chan, P. (2008), 'Learning implicit user interest hierarchy for context in personalization', *Appl. Intell.* **28**(2), 153–166.

Kobsa, A. (2007), Generic user modeling systems, *in* P. Brusilovsky, A. Kobsa & W. Nejdl, eds, 'The Adaptive Web', pp. 136–154.

Li, L., Wang, D., Li, T., Knox, D. & Padmanabhan, B. (2011), Scene: A scalable two-stage personalized news recommendation system, SIGIR, pp. 125–134.

Michlmayr, E. & Cayzer, S. (2007), Learning user profiles from tagging data and leveraging them for personal (ized) information access, *in* 'WWW', pp. 1–7.

Miller, G. A. (1995), 'Wordnet: A lexical database for english', *Commun. ACM* **38**(11), 39–41.

Orlandi, F., Breslin, J. & Passant, A. (2012), Aggregated, interoperable and multi-domain user profiles for the social web, I-SEMANTICS, pp. 41–48.

Saleheen, S. & Lai, W. (2013), User interest based complex web information visualization, *in* 'APWeb', pp. 429–436.

Saleheen, S. & Lai, W. (2014), A new type of web graph for personalized visualization, *in* 'PacificVis', pp. 238–242.

Tao, X., Li, Y. & Zhong, N. (2011), 'A personalized ontology model for web information gathering', *KDE* **23**(4), 496–511.