User Interest Model at Multiple Dimensions and Multiple Granularities

Hui Chen, Song Deng
School of Software and Communication Engineering,
Jiangxi University of Finance and Economics,
Nanchang, China,
E-mail: {chen_hui, dengsong}@126.com

***Abstract*-Finding user's interest and pushing related resources to user would be the best solution to aim the problem of "Information Overload" and "Lost in Internet". This paper designs a user interest model with multiple dimensions and multiple granularities, named UIM. In UIM, the interest concepts are kept by a hierarchical concept tree at different granularities, the history of interest are maintained with titled time window at multiple time granularities, and moreover current interest are discriminated from history ones by gradually fading the weights of interests. The results of simulations show that UIM is efficient to be maintained, can quickly extract users important interests in different time windows, and the correctness of finding user interest is prior to analogous methods.**

Keywords- user interest; meta-interest classification tree; titled time window; time decay model

I. INTRODUCTION

With the wide spread of the Web 2.0 applications, the resources in internet rapidly grow at an exponential speed, and the internet becomes one of the most important channels that people acquire information and knowledge. People enjoy the facilitation of internet, but also suffer the confusion of "information overload" and "lost in internet". It would be very helpful to proactively push the information to people who might be interested in by studying the interest of people [1].

Recently, exploring users' interests [2-5] and improving the performance of services for internet users have become one of the most popular topics among the text mining. Many methods and algorithms had been proposed by researchers to explore user interest. D. Godoy et al. [2] proposed a document clustering algorithm named WebDCC to incrementally unsupervised learn concept over Web document, and hence obtain user's interest model. R. W. White et al. [5] proposed a method to build user interest model by analyzing the context during the web activities, and based on which the short term interests can be predicted.

L. Li et al. [6] analyzed the reading history of users and present a news recommender system LOGO which integrates the long-term and short-term reading preferences of users. LOGO can help determine the news recommendation list to individual users. M.S. Reddy et [7] proposed a method to find the interest information by a process of frequent patterns extraction, and then the interests are distinguished by considering the categories of the items in a concept hierarchy. C.V. Pavan Kapanipathi et al. [8] proposed a structure named hierarchical interest graph to represent user interests and their hierarchical relationships, which is helpful to enhance the recommender system to personalize based on a varied level of conceptual abstractness.

The existing user interest models can be primarily divided into two classifications: static user model [2] and dynamic user model [6] [9]. The former can extract and represent user's basic information or behavior information by mining a static dataset, which is useful in personalized recommendation but cannot deal with the dynamicity of user interest over time. Compared to the former, a dynamic user model can capture the changed interests over time which can build more accurate profiles and be more useful when used with applications.

According to the implementation techniques are used to represent user interest, the user interest models can also be classified into four categories: traditional bag of words (BOW) based [10], concepts based [8,11], tag based [12] and topic based [6,9,13]. Additionally, semantic enrichment process [11] [8] are also used to enhance the scope of the words used to represent user interests, and to provide a prediction for new interests of the user. The time fading mechanism [9] [6] are used to differentiate the weights of current interests from those of history ones.

In this literature, a user interest model at multiple dimensions and multiple granularities (shortly UIM) is present to dynamically maintain user's interests. The contributions of this work are as follows: (1) a meta-interest classification tree (shortly MIC-tree) is proposed to hierarchically categorize the concepts of user interests, which can provide multiple granularities of conceptual abstractness; (2) a time decay model is also present to distinguish the current interests from the history ones by gradually fading the weights of interests; (3) a titled time window is present to maintain the shifting process of any meta-interest in MIC-tree, and a fine time granularity is used to kept the weight in current time window and the coarse one is used to kept the weight in history time window; and (4) the user interests at any time window can be quickly mined from UIM, thus the shift tendency of any user interest can be easily captured.

II. ANALYSIS ON USER INTEREST

In order to clearly express our user interest model at multiple dimension and multiple angularities, we first present some definitions on user interest, and then discuss the multiple granularities time window model and time decay mechanism that is used to distinguish the history interests and the current ones.

A. Meta-interest

As user visits web resources, the process of web browsing will generate web logs. The interests of users can be learned by analyzing the logs, and normally described by a set of concepts. Among of all the concepts, some are important to express user's interest, but some are not. In order to clearly express user's interests and the interest concepts, we first present following definitions.

- Definition 1: (Meta-Interest) A concept extracted from user's logs and could be used to describe user interest is called a *meta-interest* of the user, denoted by *MI*.

A meta-interest shows one respect of expectation on the information that user wants to obtain, so each meta-interest has its weight to indicate the degree of user's expectation. In a special time interval, many meta-interests can be extracted from user's logs, and the weight of each, denoted by W, can be calculated by the valid visiting time on the related web resource. Obviously, the value of W is bigger, the MI is more important for user, and vice versa.

- Definition 2: (Important Meta-Interest) Given a integer k($k>1$) and a time interval, the top-k weight meta-interests are called the *important meta-interests* in that time interval.
- Definition 3: (Secondary Meta-Interest) Given a threshold δ($0<\delta<<1$) and a time interval, the meta-interests whose weight are less than δ are called the *secondary meta- interests* in that time interval.
- Definition 4: (Time-unit) When analyzing the interest of users, a time period T is assumed to be basic and indivisible, and T is called a *time-unit*.

If the time-unit T is determined, the visiting history of user can be divided into multiple segments based on the uniform time period T . For each segment, a set of meta-interests and their weights can be extracted by analyzing the log in the segment, and it is denoted by *MIS*. So, a list of meta-interest sets can be obtained by analyzing the whole log of a user, and denoted by $MIS\text{-}List=MIS_1,MIS_2,\ldots, MIS_n$, where MIS_i is the meta-interest set of user in the i^{th} time period, and MIS_1 is the oldest meta-interest set.

B. Time Window at Multiple Granularities

As has discussed in section above, for any a meta-interest of user, lots of counters are required to keep the weights of a long-term meta-interest. For example, as shown in Figure 1, if the time-unit T is set to be 'day' and a uniform granularity time windows is used, 365 counters are needed to store the weights of a meta-interest in a year. It is rather costly for all meta-interests of all users especially those long-term ones.

For the case of efficiently keeping the history information of long-term meta- interests, without losing the correctness, a time window at multiple granularities is used. For flexibility, an granularity array of the time window $D = \{d_1, d_2,\ldots, d_m\}$ is defined, where m is the amount of granularities that used in the time window, d_i is the number of counters for the level-i granularity, and i=1 means the finest time granularity.

As shown in Figure 1-(b), if the time granularities of time window is defined by the granularity array $D= \{7, 4, 12\}$, and the time-unit T is set to be 'day'. Then seven time windows at level-1 granularity is used to keep the weights of meta-interest in the latest seven days. And every seven level-1 time windows will be merged into a level-2 time window. Analogously, four level-2 time windows will be merged into a level-3 time window, and so on. Obviously, only 7+4+12=23 counters can keep the weights of meta-interest more than a year, which can greatly cut down the memory space.

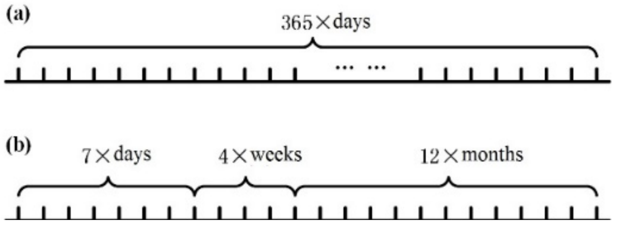

Figure 1. Sketch of time windows, (a) uniform granularity time windows, (b) multiple granularities time window.

C. Time Decay Model

Similarly, people might gently forget his history interests and become interest on new things as time passes by. So it is essential to distinguish the weights of current interests from history interests. Next we will discuss a time decay mechanism to scaling down the weight of history interest.

- Definition 5: (Decayed weight) The weight of a meta- interest is faded by the time decay model is called its decayed weight, and denoted by $\widehat{W}$.

Given a decay factor f ($0<f<1$), if the weight of a meta-interest is W_1 at the first time-unit T_1, then the decayed weight will be $W_1 * f + W_2$ at the next time-unit T_2, where W_2 is the weight in second time-unit. Similarly, the decayed weight of the meta-interest at the time-unit T_i can be calculated by the following equation.

$$\widehat{W_i} = \begin{cases} W_1 & if\ i = 1 \\ \widehat{W_{i-1}} \times f + W_i & if\ i \geq 2 \end{cases} \tag{1}$$

D. Meta-interest Classification Tree

In a long-term time period, many meta-interests can be extracted by analyzing user web logs, and the subject areas of those meat-interests might be different. In order to clearly describe the inclusion relation between those meta-interests, referring to the concept-sememe tree of HowNet [14] and the concept hierarchy tree [15], we present an improved hierarchical Meta-Interest Classification tree, simply named *MIC*-tree, to store the meta-interests of a user.

As shown in Figure 2, in MIC-tree, each node has three fields: (1) *node-name* denotes a meta-interest, (2) *childNum* registers the number of total children of the node in a standard concept hierarchy tree, and (3) *CurNum* registers the number of child of the node in *MIC*-tree. For any node, its *CurNum* is initialized to be zero and its value increases 1 as a child is inserted, and the *Childnum* field can be obtained from a standard concept hierarchy tree.

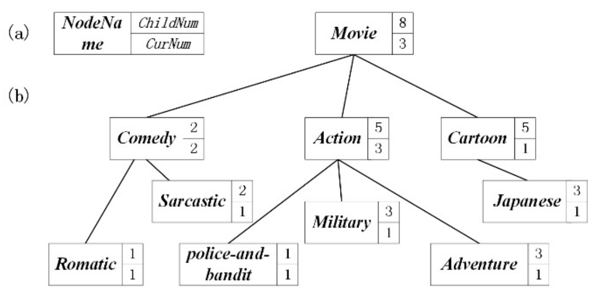

Figure 2. A meta-interest classification tree: (a) structure of node, (b) an example of tree.

Obviously, a node in MIC-tree is more abstract than its descendant on topic. In this way, the topic of user interests at multiple topic granularities can be efficiently saved on a meta-interest classification tree. Figure 2 illustrates a MIC-tree on movie. Obviously, the meta-interest ’Action’ expresses wider range topic that ’Military’ or ’Adventure’.

- Definition 6: (Interest-similar sibling) For two or more nodes with same parent, the weight ratio of any two nodes is ζ, if $|ζ−1|<γ(0<γ<1)$, then the siblings are called the interest-similar siblings.
- Definition 7: (Interest-inimitable node) A node in a MIC-tree is called a interest-inimitable node if it satisfies the following two conditions:(1) it has no child and no interest-similar sibling; (2) it has children but the children are interest-similar siblings.

As shown in Figure 2, if ’Romantic’ and ’Sarcastic’ are significant meta-interests and interest-similar siblings, then the meta-interest at coarser granularity, ’Comedy’, can be used to summarily express user’s interest on ’Romantic’ and ’Sarcastic’.

III. USER INTEREST MODEL AT MULTIPLE DIMENSIONS AND GRANULARITIES

In this section, we will present our user interest model, named *UIM*. For each user, his meta-interests are expressed by hierarchical meta-interest classification tree at multiple granularities, and for each meta-interest, its history is kept by the time window at multiple granularities.

A. Structure of UIM

As shown in Figure 3, an UIM contains two parts: (1) a *MIC*-tree, and (2) a table of Time wIndow at muLtiple granulariTies, shortly named *TILT*-table, linked to each node in the tree. In the time window, each entry has three fields: (a) identifier of time window, (b) level of time granularity as discussed above, and (c) the decayed weight of the meta-interest in that time window.

Given a time-unit *T*, the user log can be divided into a series of data segments each labeled with the identifier of relevant time window. For each data segment, a set of meta-interests can be extracted by the methods [7]. Then the meta-interests can be incrementally updated into MIC- tree of UIM as discussed next, simultaneously, the weight of each meta-interest is inserted into the TILT- Table linked to relevant node. If the number of level-1 time windows reaches its maximum, the process of merging TILT-Table in Section IV-C is conducted. Periodically, the algorithm in Section IV-D should be performed to cut down the size of UIM.

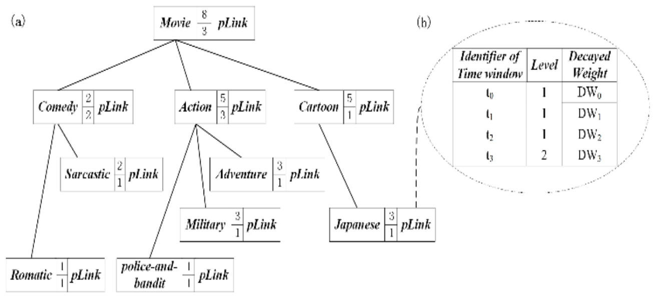

Figure 3. Sketch of an UIM: (a) MIC-tree; (b) TILT-Table of node ’Japanese’.

B. Incrementally Updating UIM

At the *i*th time-unit, a set of meta-interests, denoted *MIS*, and their weights can be obtained by analyzing the data segment in the time window T_i. Then *MIS* and the weights can be maintained by *UIM* by incrementally updating *UIM* according to Algorithm 1.

```
Algorithm 1: Incremental updating of the UIM
Input: MIS: newly extracted meta-interest, each has its weight
       $T_i$: current time window
       MI-tree: meta-interest classification tree in UIM
       TILT-table:table of time window at multiple granularities.
Output: Updated UIM with MIS being added.
1  Create a pointer TP that points to the root of UIM;
2  for $j \leftarrow 1$ to $|MIS| >$ do
3      Search the standard concept hierarchical tree to find the node labeled
         MIS[j];
4      if the node labeled MIS[j] is not found then
5          continue;
6      else
7          Get the path PATH from root to the node labeled MIS[j];
8          for Node in PATH do
9              Search the children of TP to find a node MINode with same
                 item-name as Node;
10             if MINode is found then
11                 insert a row using $(T_i, 1, W_j)$ into the TITL-table of
                     MINode, where $W_j$ is the weight of MIS[j] in $T_i$;
12                 if number of time windows at any granularity reaches its
                     maximum then
13                     Merge the TITL-table using Algorithm 2;
14                 end
15             else
16                 Create a new node MINode and insert it into MI-tree as
                     child of TP;
17                 Insert new row $(T_i, 1, W_j)$ into MINode's TITL-table;
18             end
19             Set TP pointing MINode;
20         end
21     end
22 end
```

As shown in line 3 of Algorithm 1, for any new meta-interest *MIS*[*j*] in time window T_i, we will first search a prepared concept hierarchical tree to find the node labeled *MIS*[*j*]. If not found, *MIS*[*j*] might be non-significant, or it might be new generated concept. For the latter, it could be correctly process after it is be inserted into the prepared concept hierarchical tree. If a node labeled *MIS*[*j*] is found, then the path from root to it in the tree can be obtained and denoted as *PATH* as shown in line 7. Then for each node denoted by Node in *PATH*, its weight is incrementally updated into *MIC*-tree by lines 8-20. Firstly, search the children of *TP* in the *MIC*-tree and find whether a node with the same item-name as Node exists or not. As shown in lines 10-14, suppose a node named *MINode* is found, then insert a new entry (T_i,1,W_j) into its *TILT*-table. At this time, if the number of time windows at any granularity reaches its maximum, Algorithm 2 will be conducted to merge the time windows. Or else, as shown in lines 15-18, if none of *TP*'s children matches *MINode*, then create a new node labelled *MINode* and insert it into *MIC*-tree as *TP*'s child. Also, insert a new row (T_i,1,W_j) into the *MINode*'s *TILT*-table to keep its weight. At last, set *TP* to point *MINode* and repeat the steps from line 8 to line 18 until all nodes in *PATH* are processed.

According to the process of Algorithm 1, only the paths from root to those nodes which register the meta-interests of user in prepared concept hierarchy tree are copied to build the *MIC*-tree. Thus an *MIC*-tree is far smaller than the concept hierarchy tree. Additionally, the *MIC*-tree is always updated from root to leaves when Algorithm 1 is conducted. Obviously, it is easy to derive the following properties.

- Property 1: In an *MIC*-tree, the weight (or decayed weight) of any node is no less than the sum of its children's weights (or decayed weights).

The correctness of Property 1 can be easily proved by the process of Algorithm 1. And then, two important properties can be derived from Property 1.

- Property 2: In an *MIC*-tree, if a node registers an important meta-interest, then all his ancestor must be important meta-interests too.
- Property 3: In an *MIC*-tree, if a node registers a secondary meta-interest, then all his descendant must be secondary meta-interests too.

C. Merging TILT-table

At any time, if the number of time windows at any granularity reaches its maximum, the process of merge *TILT*- table will be conducted to merge some fine granularity time windows into a coarse time window using Algorithm 2. Sometime, the process of merging time window is always conducted from the finest granularity to coarse one, and merging the time windows at lower level granularity might cause that some time windows at higher level granularity should be merged too.

Algorithm 2: Merging *TILT-table*

```
Input: $T_i$: current time window
       TILT-table: TILT-table to be merged
       $D = (d_1, d_2, \cdots, d_n)$: Array of time window number in
TILT-table.
Output: Merged TILT-table.
1  for $j \leftarrow 1$ to $n$ do
2      Get the oldest identifier $T_s$ at level-$j$ granularity in TILT-table;
3      set total=($T_i - T_s$)/$d_j$;
4      if total > 0 then
5          for $l \leftarrow 1$ to total do
6              Merge the time windows from $T_s + (l-1) \times d_j$ to
                 $T_s + l \times d_j$ into a time window at level-($j$+1) granularity
                 which is identified with $T_s + (l-1) \times d_j$;
7              Replace the time windows from $T_s + (l-1) \times d_j$ to
                 $T_s + l \times d_j$ with the new time window;
8          end
9          Set $T_s = T_s + d_j$;
10     else
11         break;
12     end
13 end
```

As shown in lines 3-4 of Algorithm 2, before merging the time windows at level-*j* granularity, we first get the identifiers of the oldest and latest time window at level- *j* granularity, which are respectively denoted as T_s and T_e. Then the times that time window at level-*j* granularity should be merged can be easily calculate by formula *total*=(T_i−T_s)/d_j, where d_j is the maximum number of time window at level-*j* granularity. If *total* is bigger than zero, then the process of merging time window should be conducted *total* times as shown in lines 6-10. When merging the window in level-*j* granularity at the l^{th} time, the time windows from $T_s+(l-1)\times d_j$ to $T_s+l\times d_j$ will be merged into a time window in level-(*j*+1) granularity and identified with $T_s+(l-1)\times d_j$. Secondly, the rows of $T_s+(l-1)\times d_j$ to $T_s+l\times d_j$ are removed from *TILT*-table and an entry about the merged time window is inserted into *TILT*-table at the correct position.

D. Pruning UIM

As more logs are processed, more meta-interests and their weights are updated into UIM, which makes the size of UIM continually increasing and difficult to maintain the model.

As has discussed in Section II, the user's interests might be shifted as time pass by. For a history meta-interest, its weight might be decayed gradually till it becomes a secondary one. Therefore, there will be more and more secondary meta-interests in an UIM. Additionally, lots of secondary meta-interests will also be inserted into UIM when analyzing web logs. Pruning the secondary meta-interests from UIM could greatly cut down the size of user model but have negligible influence on the mining correctness.

At time T_i, pruning *UIM* is the process of traversing *MIC*- tree starting from the root. When visiting any node in *MIC*- tree, firstly calculate the decayed weight of the node at time T_i according to Equation 1. If the decayed weight is less than $\delta(0<\delta<1)$ then the subtree root at the node can be safely deleted according to Property 3.

IV. MINING TOP-K META-INTERESTS

In a recommender system, it is important to find the resource which match the most significant interests of user. Even though a pruning algorithm is conducted to delete the secondary meta-interests from UIM, there are still many meta-interests in the model. In this section, a simple method will be present to mine the top-k meta-interests from UIM.

Given a integer $k(k > 0)$ and a observed time window *TW*, the top-*k* meta-interests can be mined from *UIM* by Algorithm 3. Initially, an array named *buffer*[*m*] is defined to keep the important meta-interests in UIM, where *m* is much bigger than *k*. Each element in *buffer* has three fields: *MI*, *parentMI* and *DW*, respectively denotes a meta-interest in *UIM*, its parent meta-interest and its decayed weight in time window *TW*. Then mining algorithm can be divided into three steps. As shown in lines 2-7, the first step is to traverse the *UIM*, and add all leaves into *buffer*. As shown in lines 8-18, the second step is to find the similar-interest brothers. If there exits similar-interest brothers, then replace them with their parent. After that, all meta-interests in *buffer* are Interest-inimitable node. Then, the top-*k* meta-interests can be easily obtained as shown in lines 19-20.

Algorithm 3: Mining top-*k* meta-interests

```
Input: k: number of meta-interests to be mined
       UIM: user interest model to be updated
       TW: time interval in which the top-k meta-interests will be mined
Output: top-k meta-interests.
1  define an array buffer[m]($m \gg k$);
2  create a pointer TP that points to the root of MIC-tree and traverse the tree;
3  while TP is not null do
4      if TP points a leaf then
5          add(TP->MI, TP->parentMI,DW) into buffer, where DW is
             calculated by Equation 1;
6      end
7  end
8  define a boolean flag=true;
9  while flag do
10     set flag=false; check buffer to find the elements who are brothers;
11     for each group of elements who are brother do
12         if the elements are similar-interest brother then
13             insert their parent into buffer;
14             remove those elements;
15             set flag=true;
16         end
17     end
18 end
19 order the elements in buffer in DW descendant order;
20 output the top-k meta-interests in buffer;
```

V. PERFORMANCE EVALUATION

In this section, the experimental setup is firstly described, then the results of performance study are presented.

A. Experimental Setup

All of the experiments were performed on a PC with Intel I7-4500 CPU and 8GB of main memory running Windows 7. All programs were written in C++. About 8.9 million book reviews data [16] collected from Amazon.com during May 1996 to July 2014 were used in the experiments. In the experiments, all meta-interests of users were obtained as preprocess was conducted, the cost of which was excluded from those of the experiments. The default value of decay factor is set to be 0.98 if there is no special declaration.

B. Performance and Comparison Results

Firstly, two sets of experiments are conducted to evaluate the correctness of UIM as decay factor f varies. In the experiments, the value of f varies from 0.965 to 0.99, then we analyze the recall of mining top-*k* (*k*=10) in the future time window of 10 and 30 days.

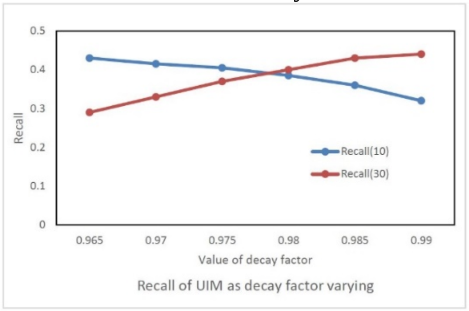

Figure 4. Correctness of UIM as decay factor varies

As shown in Figure 4, when the value of decay factor increases, the correctness on mining the future time window of 10 days decreases tardily, but that of 30 days is gracefully increasing. This is because the weight of an important meta-interest will be fast decayed if a smaller f used, and then an important meta-interest might change to be a secondary one. That is also to say, it is helpful to mine long-term meta-interests if smaller value of f is chosen.

Secondly, three sets of experiments are conducted to analyze the memory usage of UIM. For the purpose of comparison, we implements three user models: (a) UIM, the present model; (b) UIM-1, the UIM using single time granularity; and (c) UIM-2, the UIM without pruning process.

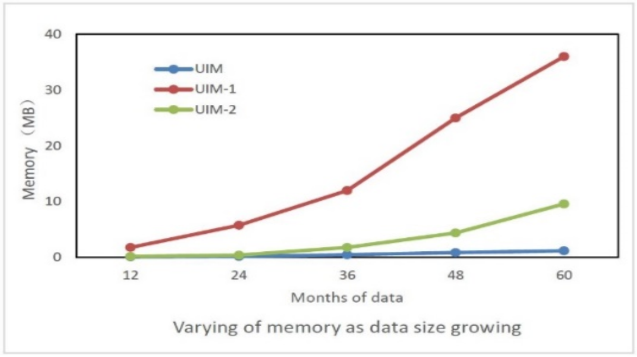

Figure 5. Cost on memory of UIM as size of time window varies.

In the experiments, the finest time granularity is set to be 'day', and the memory usage of three models as size of time window varies are shown in Figure 5. Obviously, our UIM is prior to the other two models. As the size of time window increased, much more counters are necessary to keep the detail weight of any meta-interest for UIM-1, so its space is much costly compared to other two models. UIM-2 keeps all meta-interests including secondary ones, so its space will gracefully increase as more secondary meta-interests are updated into the model as size of time window grows.

Thirdly, the correctness of UIM on mining top-*k* meta-interests are analyzed as the size of time window varies. In the experiments, *k* is respectively set to be 10, 20 and 30. In the experiments, the data in one year are selected to build our interest model, and the data in following month are used to test the correctness of UIM.

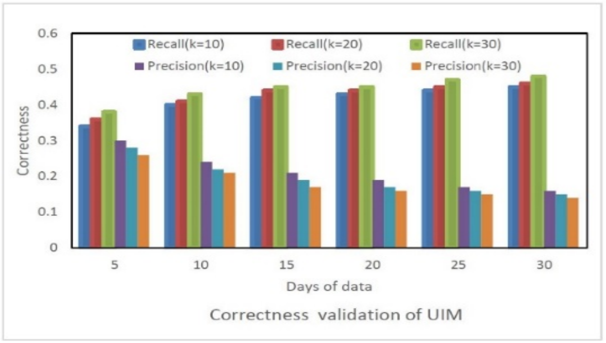

Figure 6. Correctness of UIM as the value of k and size of time window varies.

As shown in Figure 6, when mining top-k meta- interests in the any sized time window, the recall of a bigger k is a little greater than that of a smaller one, but the tendency of precision is reversed. Similarly, the recall is increasing but precision is decreasing as the size of time window grows. If k is bigger, more meta-interests will be mined, which will be help to find steady and long term interests. Similarly, a big sized time window is focused when mining UIM, the recall will increase because a newly short-term interest might be outstanding in a short time window but unambitious in a long time window.

Lastly, three sets of experiments are conducted to com- pare the correctness of the proposed model and LOGO [6] and TimeFM-user [9]. In the experiments, the data in one year are similarly selected to build interest models, and the data in following month are used to test the correctness of three models. The recalls and precisions of three methods on mining top-k are analyzed, where k is set to be 10.

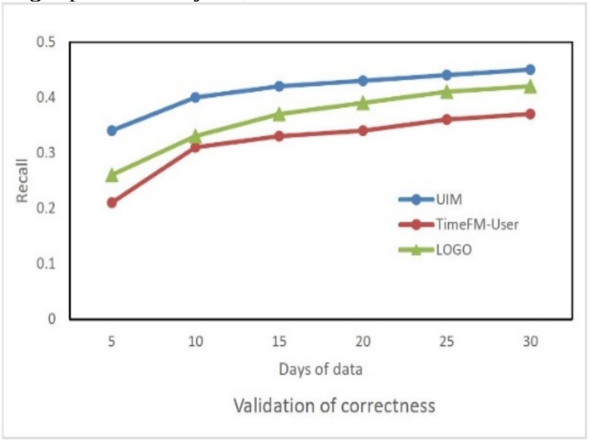

Figure 7. Recall of three methods as size of time window varies

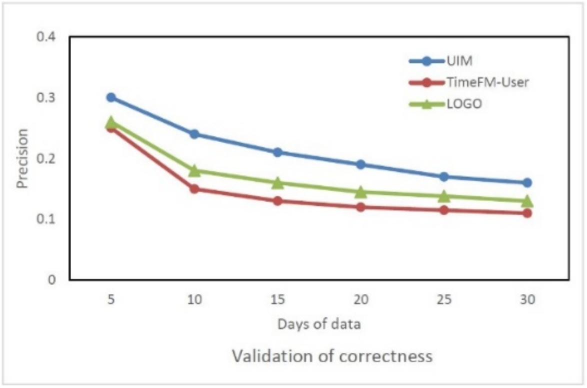

Figure 8. Precision of three methods as size of time window varies.

As shown in Figure 7 and 8, the correctness of proposed model is the best on mining top-10 user interests. The main reason might be analyzed as follow. The TimeFM-user method can differentiated the weights of current and history interests by fading mechanism, however only the finally faded weights from the beginning can be obtained. When mining the interests in test data, the interests in training data cannot be excluded, which will the correctness of mining results. As for the LOGO method, a similar time decay mechanism is also used. Additionally, the groups of topics that user refers are regarded as long-term interests, and the preferred items in a group are regarded as short- term interests. Which is not accurate when finding the user interests. Compared to the other two methods, the proposed method accurately maintains the meta-interests with concept hierarchy tree and weights of meta-interests with titled time window. When mining the interests in a certain time window, the meta-interests and their weighs in the time window can be demarcated from others. Thus the correctness of mining results of the present methods is certainly the best among of the three methods.

VI. CONCLUSION

This paper proposed a user interest model named UIM to incrementally maintain the meta-interests of user and their weights. UIM can incrementally maintain user's interests and their weights at multiple granularities. Experimental results show that UIM is efficient to solve the problem of “information overload” and “lost in Internet”, and its correctness is prior to analogous methods.

ACKNOWLEDGEMENT

This work was partly supported by the National Science Foundation of China under grant No. 61262033, 61462037; the Natural Science Foundation of Jiangxi Province, China under grant No.20142BAB207009, 20142BAB217014.

REFERENCES

[1] A.-H. Ahmad and X. Yue, “A survey of user modelling in social media websites,” Computer and Information Science, vol. 6, no. 4, pp. 59–71, 2013.

[2] D. Godoy and A. Amandi, “Modeling user interests by conceptual clustering,” Information Systems, vol. 31, no. 4-5, pp. 247–265, June-July 2006.

[3] X. Liu, Y. Liu, and K. Aberer, “Personalzied point-of-interest recommendation by mining users’ preference transition,” in Proceedings of the 22nd ACM International Conference on Information and Knowledge Management. San Francisco, CA, USA, October 27- November 1 2013, pp. 733–738.

[4] Y. Liu, C. Liu, B. Liu, and etc., “Unified point-of-interest recommendation with temporal interval assessment,” in Pro- ceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Francisco, CA, USA, August 13-17 2016, pp. 1015–1024.

[5] R. W. White, P. N. Bennett, and S. T. Dumais, “Predicting short-term interests using activity-based search context,” in Proceedings of the 19th ACM international conference on In- formation and knowledge management, CIKM’10. Tornoto, Canada, October 26-30 2010, pp. 1009–1018.

[6] L. Li, L. Zheng, and T. Li, “Logo: A long-short user in- terest integration in personalized news recommendation,” in Proceedings of the fifth ACM conference on Recommender systems, RecSys’11. Chicago, IL, USA, October 23-27 2011, pp. 317–320.

[7] M. S. Reddy and S. Srivastava, “Extracting diverse patterns with unbalanced concept hierarchy,” in Proceedings of 18th Pacific-Asia Conference, PAKDD 2014. Tainan, Taiwan, May 13-16 2014, pp. 15–27.

[8] C. V. Pavan Kapanipathi, Prateek Jain and etc., “User interests identification on twitter using a hierarchical knowledge base,” in Proceedings of 11th International Conference, ESWC 2014. Anissaras, Crete, Creece, May 25-29 2014, pp. 25–29.

[9] X. Tang, Y. Xu, and S. Geva, "Integrating time forgetting mechanisms into topic-based user interest profiling," in Pro- ceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology(2013). Altanta, GA, USA, November 17-20 2013, pp. 1–4.

[10] J. Hannon, M. Bennett, and B. Smyth, "Recommending twitter users to follow using content and collaborative filtering approaches," in Proceedings of the fourth ACM conference on Recommender systems, RecSys'10. Barcelona, Spain, October 26th-30th 2010, pp. 199–206.

[11] C. Lu, W. Lam, and Y. Zhang, "Twitter user modeling and tweets recommendation based on wikipedia concept graph," in Workshops at the Twenty-Sixth AAAI Conference on Ar- tificial Intelligence. Toronto, Ontario, Canada, July 22-26 2012, pp. 33–38.

[12] J. Hannon, K. McCarthy, M. P. OMahony, and B. Smy, "A multi-faceted user model for twitter," in Proceedings of the 20th International Conference on User Modeling, Adaptation, and Personalization, UMAP 2012. Montreal, Canada, July 16-20 2012, pp. 303–309.

[13] L. Li, L. Zheng, F. Yang, and T. Li, "Modeling and broad- ening temporal user interest in personalized news recommen- dation," Expert Systems with Applications, vol. 41, no. 7, pp. 3168–3177, 2014.

[14] B. Wu, J. Yang, and L. He, "Chinese hownet-based multi- factor word similarity algorithm integrated of result modifi- cation," in Proceedings of International Conference on Neural Information Processing. Doha, Qatar, November 12-15 2012, pp. 256–266.

[15] D. Tu, L. Chen, and G. Chen, "Automatic multi-way domain concept hierarchy construction from customer reviews," Neu- rocoputing, vol. 147, pp. 472–484, January 2015.

[16] R. P. Julian McAuley and J. Leskovec, "Inferring networks of substitutable and complementary products," in Proceedings of the 21th ACM SIGKDD International Conference on Knowl- edge Discovery and Data Mining, KDD'15. Hilton, Sydney, Austin, Auguest 10-14 2015, pp. 785–794.