Journal of Universal Computer Science, vol. 15, no. 12 (2009), 2387-2408
submitted: 15/12/09, accepted: 25/6/09, appeared: 28/6/09 © J.UCS

A Joint Web Resource Recommendation Method based on Category Tree and Associate Graph

Linkai Weng, Yaoxue Zhang, Yuezhi Zhou
(Key Laboratory of Pervasive Computing, Ministry of Education
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Science & Technology
Tsinghua University, Beijing, China
wlk02@mails.tsinghua.edu.cn, zyx@moe.edu.cn
zhouyz@mail.tsinghua.edu.cn)

Laurence T. Yang
(Department of Computer Science, St .Francis Xavier University
Antigonish, Canada
ltyang@gmail.com)

Pengwei Tian, Ming Zhong
(Key Laboratory of Pervasive Computing, Ministry of Education
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Science & Technology
Tsinghua University, Beijing, China
{tpw04, zhong-m}@mails.tsinghua.edu.cn)

Abstract: Personalized recommendation is valuable in various web applications, such as e-commerce, music sharing, and news releasing, etc. Most existing recommendation methods require users to register and provide their private information before gaining access to any services, whereas a majority of users are reluctant to do so, which greatly limits the range of application of such recommendation methods. In the non-register environments, the only available information is the content or attributes of resources and the click-through chains of user sessions, so that many recommendation methods fail to work effectively due to the rating sparsity [Adomavicius and Tuzhilin, 2005] and illegibility of user identity, collaborative filtering [Goldberg et al. 1992] is an example of this case. In this paper we propose a joint recommendation method combining together two approaches, namely the domain category tree and the associate graph, to make full use of all available information. Further, an associate graph propagation method is designed to improve the traditional associate filtering method by integrating additional graphical considerations into them. Experiment results show that our method outperforms either the single category tree approach or the single associate graph approach, and it can provide acceptable recommendation services even in the non-register environment.

Keywords: personalized service, personalized recommendation, category tree, graph propagation
Categories: L.1.3, L.2.2, M.4, M.5

1 Introduction

With the increased popularity of ubiquitous applications, service personalization and user interaction have been widely accepted and used by people of all levels. In recently years, the rapid development of web technologies and exponential increase of web resources have brought people’s attention to web application, which is an important type of ubiquitous applications, since traditional internet services can’t satisfy users‘ requirements any more. Web applications are currently changing from resource-centered towards user-oriented along with the popularization of web 2.0 and the emergence of various personalized services, such as active service [Zhang and Fang, 2005].

Up to now, numerous web resources are overspread on the internet; to cite some figures, 2.5 million books are provided in Amazon (www.amazon.com) and nearly 3 million papers are available in Libra (libra.msra.cn). Among such a vast sea of information, obtaining the needed web resources can be an extremely difficult and time-consuming task. Search engines can help accomplish the task to some extent, but they are insufficient due to the following two facts: 1) they are in lack of personalized consideration; 2) they are hard to specify a proper query to precisely describe users’ concrete interests, since users are totally blind about the overall situation of all resources. To solve these problems, personalized recommendation are proposed, in which web resources can be actively pushed to those interested users based on user interaction and personalized considerations.

Personalized recommendation plays a significant role in modern information obtaining and releasing, which is why it has attracted much attention from both research and industrial societies. However, most recommendation methods require users to register, but most users are reluctant to do so due to such consideratons as time saving and privacy protection. This contradiction limits the wide spread of recommendation systems, so a method with acceptable recommendation performance in the non-register environment would surely have significant impact on the popularization of recommendation applications.

In the non-register environment, only the content or attributes of resources and the click-through chains of user sessions are available, which leads to rating sparsity and illegibility of user identity, so that most existing recommendation methods prove insufficient and ineffective. Rule-based filtering, which pushes items according to certain rules specified by users, makes it necessary for users to specify the rules each time (as no user profile records the historical rules), and they would surely be tired of this in a short time. Associate filtering [Brin et al., 1997], the method to push related items based on a co-purchasing relationship between them, can be an effective method as each click-through chain of a user session could be treated as a purchase list, but the click-through records of the current user is quite short due to the shortness of each session, so this method needs further improvement to dig out deeper co-purchasing relations. Content-based filtering [Mostafa et al., 1997], which pushes items by content similarity, seems an ideal method here, but its new interest discovery problem [Adomavicius and Tuzhilin, 2005] is always a bottleneck of the system performance. Collaborative filtering [Goldberg et al., 1992], a method to push items by considering collaborative users’ ratings of this item in the user-rating matrix, would be quite ineffective here because of rating sparsity and cold start [Adomavicius and Tuzhilin, 2005].

In this paper, we propose a novel joint method for web resource recommendation based on the domain category tree, a structural content-based filtering, and associate graph, an enhanced associate filtering. Firstly, we generate an ontology-like item category tree to make fully use of the content and attribute of relevant items of web resources, which can improve the traditional content-based method with the help of tree-related knowledge. Secondly, to tap the full potential of limited user data, we organize those resource items into an associate graph based on the click-through chains of user sessions, upon which a special adaptive graph propagation approach is designed. This method is based on graph structure and thus it can borrow the state-of-the-art from graph-related research area to dig out deeper co-purchasing relationship among items compared with traditional associate methods. Finally, considering that only content feature is used in the category tree method and only statistical social feature is used in the associate graph method, we integrate the two methods into a joint one, which could help compensate the deficiency of using either method alone, such as new interest discovery problem [Adomavicius and Tuzhilin, 2005], rating sparsity and cold start [Adomavicius and Tuzhilin, 2005], and could indeed provide acceptable recommendation services even in a non-register environment.

The remaining part of this paper is organized as follows. Section 2, tries to formulize the recommendation problem and introduce related work in this fields. Section 3, continues to introduce the framework of our joint recommendation method. After that, the domain category tree method, the associate graph propagation method and the joint method integrating these two together are described respectively in Section 4, 5 and 6. Section 7 explains a set of practical experiments that have been conducted and analyzed their results. And the last section, summarizes our conclusions and our future plans in this fields.

2 Related Work

Generally speaking, recommendation is a process to push appropriate items to certain potential interested users. More formally, the task can be formulized as follows [Adomavicius and Tuzhilin, 2005]:

$I = \{i_1, i_2, i_3, \cdots, i_m\}$, the set of all items that can be recommended,

$U = \{u_1, u_2, u_3, \cdots, u_n\}$, the set of all users who would be potential receivers,

Let fav be a favorite function denoting the usefulness of item *i* to user *u*

$$fav: I \times U \to S,$$

in which S denotes the set of scores which can be assigned to the items.

Now recommendation can be defined as a process to push items selected from *I* with high score through the favorite function fav to a certain user *u*, and it can be formulized as:

$$\forall u \in U, \qquad i'_u = \arg\max fav(i, u), i \in I .$$

The numerous challengs to be solved and potential commercial values related with recommendation have brought about great interest and enthusiasm from both the research society and some industrial players. Up to now, recommendation methods could be grouped into five primary categories along with its developing course: rule-based filtering, association filtering [Brin et al., 1997], content-based filtering [Mostafa et al., 1997], collaborative filtering [Goldberg et al., 1992], and hybrid method [Srivastava et al., 2000].

Rule-based filtering is the earliest recommendation method, which filters items with certain rules specified by users. For example, in case a user specifies a rule that items at a price higher than 50$ are unacceptable, items with higher prices would be filtered away. This method is relatively simple and effective, but the problem is that in most situations, users may feel difficult to specify their own rules. And worst of all, these rules need to be revised frequently when time or situation is changed, which is definitely a disaster to users, especially in the non-register environment where user profile recording historial user rules is absent.

Associate filtering [Brin et al., 1997] is a traditional recommendation method for commercial goods with a long history of application in supermarkets. The idea is quite simple: if item A and item B are bought together by users frequently, then when a user buys item A, item B would be recommended to this user. The well-known beer and diaper case [Girard, 2008] is a typical associate recommendation example. Nowadays, this idea is also borrowed and conducted in the web recommendation applications. For example, [Pohl et al., 2007] uses co-downloading relationship based on digital library access records in recommending research papers, which as they claimed, outperforms the author co-citation mining method in recommending newly-published papers. Associate filtering is quite suitable to our non-register scenarios, but it needs improvement for deeper mining as the user data of the current session is insufficient.

Content-based filtering [Mostafa et al., 1997] mainly considers the content features of items to select out appropriate items. This matching process can be conducted between item and item or between item and user. It may be the most active field in the recommendation research society, since it is always closely related with the hottest outcomes in the information retrieval and machine learning areas, such as dimension reduction [Jolliffe, 2002; Pedro and Pazzani, 1997], topic extraction [Hofmann, 1999; Blei et al., 2003] and natural language process. There are also various content-based systems, such as Syskill & Webert [Pazzani et al., 1996], Personal WebWatcher [Mladenic, 2000], CiteSeer [Bollacker et al., 2000], PVA [Chen et al., 2001], Pandora (www.pandora.com), etc. Content-based filtering seems another ideal method for the non-register scenarios, but its new interest discovery problem [Adomavicius and Tuzhilin, 2005] is always one bottleneck of the system performance.

Collaborative filtering [Goldberg et al., 1992], which tries to push an item to particular users based on the ratings of other similar users about this item, may be today‘s most popular recommendation method. In modern recommendation applications, the number of users is often larger than the number of items, and the number of items is rather static compared with that of users. Therefore, item-based collaborative filtering [Sarwar et al., 2001; Linden et al., 2003] emerges as application demands, which pushes related items to a certain user based on her/his ratings of other similar items in the user-rating matrix. This method, which treats the recommendation problem from the item point of view rather than the user point of view, can indeed reduce the system's dimension from user number to the relatively smaller item number and turn expensive online computation into relatively low-cost offline computation. Correspondingly, the previous user-oriented methods are called user-based collaborative filtering. Many types of systems emerged based on this category of filtering method, such as Grouplens [Resinick et al., 1994], Firefly [Shardan and Maes, 1995], WebWatcher [Joachims, 1997], Douban (www.douban.com) and Amazon (www.amazon.com), to name just a few. Collaborative filtering achieves great success in commercial recommendation systems, but it also has its own pitfalls such as rating sparsity, cold start and scalability [Adomavicius and Tuzhilin, 2005], with the former two being especially severe in the non-register scenarios.

Hybrid methods [Srivastava et al., 2000] refer to the methods combining multiple features together to provide better recommendation services. Various methods can be categorized into this group: [Claypool et al., 1999] puts foward a linear scheme to combine ratings from content-based filtering and collaborative filtering together in the online newspaper recommendation; similarly, [Pazzani, 1999] puts forward a voting scheme to combine the above two together; [Melville et al., 2002] mainly use an collaborative filtering in which the rating matrix is complemented by the ratings calculated with content-based method to relieve the sparsity problem; [Basilico and Hofmann, 2004] puts forward a joint framework combining content and collaborative filtering together with kernel function. Hybrid methods usually have better effect because more features are taken into consideration and they can complement each other to a certain extent. But as a side effect, in most cases such methods may bring about high computational costs. It is worth noting that, most existing hybrid methods combines the content-based filtering and collaborative filtering, which needs user ratings and fails to work in our non-register scenario.

The method proposed in this paper is a novel hybrid method, which combines the domain category tree matching (an improved content filtering) and the associate graph propagation (an improved associate recommendation) together. Domain category tree has wide applications in ontology-based search and faceted search [Tunkelang, 2006; Koren et al., 2008], and graph propagation [Zhu and Ghahramani, 2002], also called rating propagation, is an approach proposed in the semi-supervised learning, which is also adopted in the recommendation scheme nowadays [Huang et al., 2002; Zhou et al., 2005]. Our method takes on the advantages of the above two approaches, such as hard-discovery of new interest, and overcomes the shortages like rating sparsity and cold start. As a result, our method may well provide acceptable recommendation services in the non-register environment, which obviously helps to accelerate the popularization of recommendation services.

3 Framework of Our Joint Recommendation Method

A general personalized recommendation framework often consists of five main modules: resource organization, user modeling, item-user matching, evaluation and weight training, as is shown in Figure 1.

Resource organization module is used to describe items with a general format and to build items into a proper structure to fully utilize the relationship among them. User modeling is the module to analyze the user data to extract a unify user representing, which reflects users' specific interests. After the resource description and user model establishment, we use the item-user matching module to select users' potential interested items with certain matching and filtering methods. Evaluation module is designed to evaluate the system's effect, which also helps to train the parameters in the parameters training module.

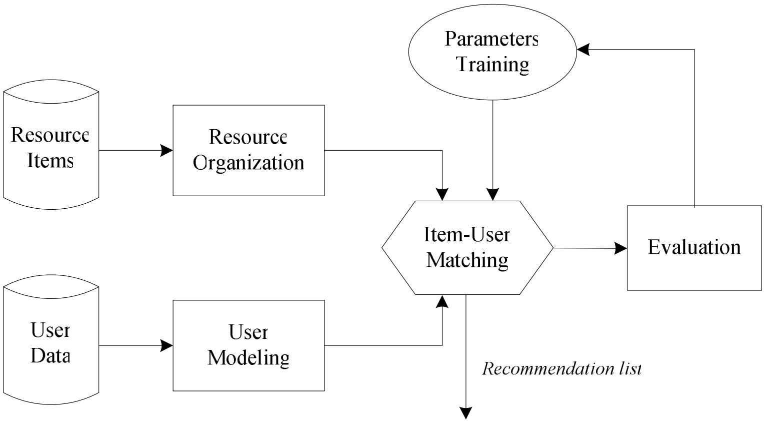


Figure 1: General Recommendation Framework

Most existing recommendation methods require users to register to get user profile, which severely limits the wide spread of recommendation applications. To provide effective recommendation services in the non-register environment, in which only limited user data is available, we proposed a joint recommendation method combining the domain category tree method and the associate graph method together to make full use of the available data.

The following three sections will, respectively, introduce the domain category tree method, the associate graph propagation method and the joint method combining the two together.

4 The Domain Category Tree Method

Domain category tree, which can be seen as an adaptive content-based filtering, has been widely adopted in various fields, such as semantic web and ontology-based search. We apply this method in our recommendation scenario to reveal more profound relations among items except the traditional content-similar relations with the help of ontology-like tree.

4.1 Creation of Meta Data Domain Category Tree

Resource descriptive format mainly consists of two parts: content and attribute. Content refers to the original part of the resource item, which can be used as the content feature in the recommendation process, for example, a paragraph of article could be processed into a language vector, and a piece of music could be decomposed into a syllable vector. Attribute is the extra descriptive data for an item, sometimes called meta-data. It is quite useful in building the domain category hierarchical tree and roughly filtering items. For example, in the book case, attributes may include ISBN, author, year, publisher, and in the courseware case, they may include subject, department, school, teacher, and year.

To make full use of the content and attribute information, we organize resource items into a domain category tree.

Definition 1: ***domain category tree***

Domain category tree is a hierarchical tree which organizes all resource items together with their content and attribute information. It can be formulized as:

$$I_{tree} = Tree(nodes, links),$$

in which each middle node represents a category of certain domain; each leaf node represents an item of resource, and link represents the parent-child relationship in the tree.

Each node can be formulized as:

$$Node = < (a_1, nv_1), (a_2, nv_2), \cdots, (a_n, nv_n) > ,$$

in which *a* represents an attribute, and *nv* represents the corresponding value of that attribute.

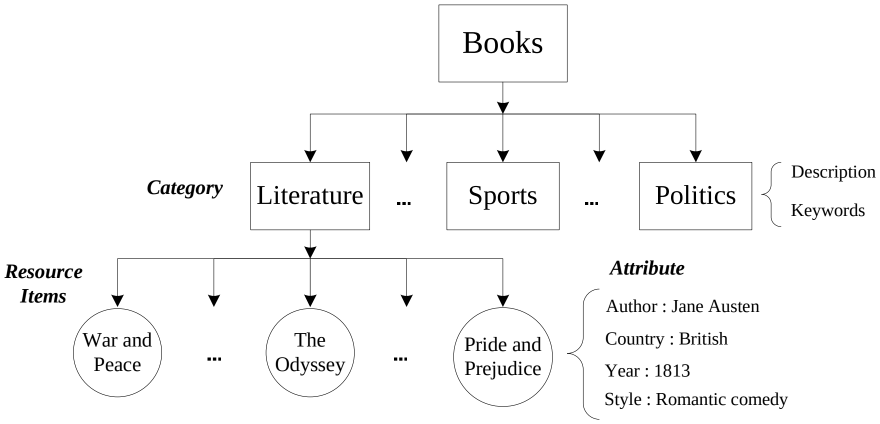


Figure 2: An Example of Domain Category Tree

Figure 2 is an example of domain category tree for the case of books, in which the middle nodes, such as “literature”, “sports” and “politics”, denote certain categories of books, and the leaf nodes, such as “War and Peace”, “The Odyssey” and “Pride and Prejudice”, show that these specific books belong to the category of “Literature”. Each node in the tree is represented as an attribute vector, for example, *Node* ("Pride and Prejudice") =<(Author, Jane Austen), (Country, British), (Year, 1813), (Style, Romantic comedy)>.

Two methods can be used to build the tree: content classification and attribute organization.

Content classification means to build the category tree by classification methods. Nearly all current recommendation applications focus on certain domain, so it is feasible to get a pre-defined domain category tree for certain applications, such as the category tree for books illustrated in Figure 2. Then we can train a classification model to assign each item to the corresponding node in this book category tree. Lots of classification methods can be applied here, such as Naïve Bayes Classifier [Domingos and Pazzani, 1997], K-Nearest Neighbor Algorithm [Shakhnarovich et al., 2005] and Support Vector Machine [Cortes and Vapnik, 1995]. Although this method needs additional manual supervising effort, its effect is quite satisfying.

Content classification method will fail to work when there is no content of items in hand or the application requires less manual efforts. Attribute organization provides a feasible solution to this, which builds the category tree based on the attributes of each item. For example, in recommending books, we can categorize books into groups using its author attribute, and thus the books of the same author will appear under the same author node of the tree, which means one author's all works would be recommended via each other. Further, the node of the tree often has practical meaning and has some sub-attributes. In the above example concerning books, the author node could have its own sub-attributes, namely gender, nationality and style, all of which would be useful when filtering items by similarity comparison. Attribute organization method is an effective resource organization method especially suitable for the situation in which only attributes are available.

4.2 User Attribute Modelling

User modeling is the key part in the entire recommendation process. In our non-register scenario, the only available data concerning user is the usage behavior in the current session. Usage behavior refers to all the activities of a user in using the system, including selected items, time duration, submitted query, and submitted remark, etc, all of which could be used to reveal the user's implicit interests.

We construct the user interest model based on the usage behavior. By processing user behavior records carefully, we can make a list of a user's favored items. For example, if a user downloads an item, or views an item for a long time, we think the user would be interested in this item. This list of favored items is represented as:

$$u \equiv favList = \{ i_1, i_2, \cdots, i_n \} = \{(i_1, v_1), (i_2, v_2), \cdots, (i_n, v_n)\},$$

each pair within the parentheses represents a favored item, in which *i* is the item name and *v* is the corresponding favor value.

Now we can construct the user attribute model based on the above list of the user's favored items. Each item *i* in the list has its own attribute vector $< (a_1, nv_1), (a_2, nv_2), \cdots, (a_n, nv_n) >$, and for each attribute *a*, by merging the corresponding value nv of every *i* together with the weight *v*, we can get this attribute's user value. Then the final user attribute model would be $u_a =< (a_1, uv_1), (a_2, uv_2), \cdots, (a_n, uv_n) >$, which is the user model in the domain category tree method.

4.3 Domain Category Tree Matching

The matching process consists of two stages: matching stage and diffusing stage.

·Matching stage

This involves matching certain users with all nodes in the tree one by one, which can be formulized as:

$$S_{tree} = fav_{tree}(node, u_a, w_{tree})$$

$$= w_{a1} \times fav_{a1}(a_1, nv_1, uv_1) + \cdots + w_{an} \times fav_{an}(a_n, nv_n, uv_n)$$

·Diffusing stage

In this stage, the user‘s interested items would be diffused though the category tree as all items have a kind of structural relationship with each other. Therefore, after getting the matching score of each node, we diffuse the score S_m through tree link, mainly parent-children relationship and sibling relationship.

Figure 3 is an example of domain category tree matching still with the case of books as the scenario. At first, the user attribute model is represented as < (Favored author, Jane Austen), (Nationality, Chinese), (Favored year, Modern), (Favored style, Romantic poem), (Acceptable price, low) >. Then the user model is compared with leaf nodes in the tree one by one. For example, the “Pride and Prejudice” is completely matched in “author” attribute and partly matched in “style” attribute. And finally, the matching score would be diffused through the relationship between the nodes in the tree, mainly the parent-child relationship (node *a* and node *b*) and the sibling relationship (node *b* and node *c*).

The above domain category tree matching process is described in detail in Algorithm 1.

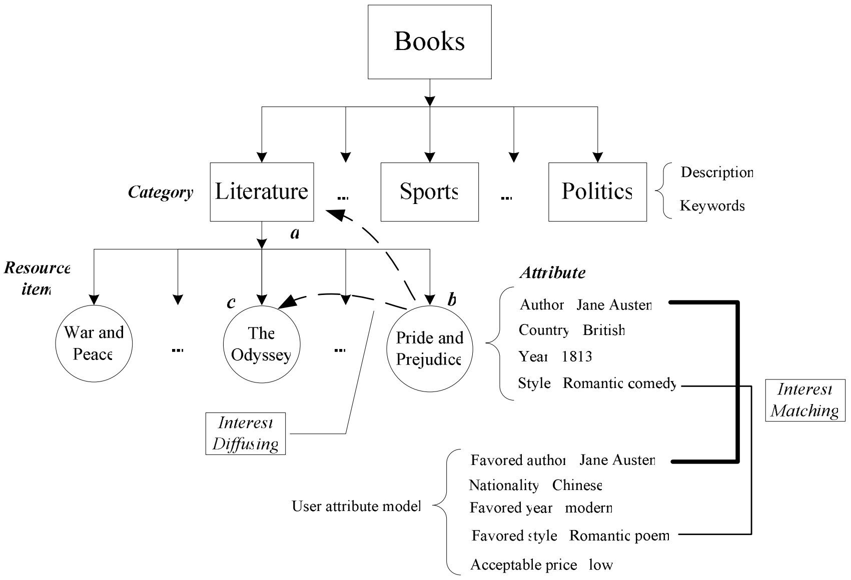


Figure 3: An Example of Associate Graph

5 The Associate Graph Method

In this section, we would discuss the associate graph method, an enhanced associate filtering suitable for non-register scenarios. In the non-register scenario, the user data collected concerning the current user is relatively insufficient due to the shortness of one session, and thus the traditional associate method can't excavate enough associate items. Our method organizes the associate items into a novel associate graph, based on which a graph propagation approach is designed to dig out deeper associate relationships between these items than those that can be excavated in traditional methods.

Algorithm 1: *Domain Category Tree Matching*

```
Input: $\mathbf{I}_{tree} = Tree(nodes, links)$, a category tree
        $node =< (a_1, nv_1), (a_2, nv_2), \cdots, (a_n, nv_n) >$ , each node in the tree
        $u_a =< (a_1, uv_1), (a_2, uv_2), \cdots, (a_n, uv_n) >$ , a user attribute model
        $w_a =< (a_1, w_1), (a_2, w_2), \cdots, (a_n, w_n) >$ , weight vector
        $\mathbf{M}_w = \begin{pmatrix} w_{11} & \cdots & w_{1n} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nn} \end{pmatrix}$, propagation weight matrix
Output: $S_{tree} = (s_1, s_2, \cdots, s_n)$, the match score of each node for the specific user
Define:
  begin
    //matching stage
    for each $node_k$ in the category tree $\mathbf{I}_{tree}$ do
      for each $(a_i, nv_i)$ in the node vector $node_k$ do
        for each $(a_j, uv_j)$ in the user attribute model do
          If $a_i == a_j$ then
            $s_k = s_k + w_i \times$ match ($nv_i$, $uv_j$)
        //match is the function comparing two values
        end for.
      end for.
    end for.
    //diffusing stage
    $S_{tree} \coloneqq \mathbf{M}_w^n \times S_{tree}$ , // n is the diffusion iteration times
  end.
```

5.1 Associate Graph

Besides the co-category relationship constructed via the content or attribute similarity, there are still other latent relations between items. Associate relation is one of them, which is the backbone of traditional associate recommendation. We declare two items have an associate relationship when they are favored by the same user, and the more users they are co-favored, the stronger such associate relationship is. In this method, we organize all the associate relations into a global associate graph, in seeking for deeper mining compared with traditional associate recommendation.

Definition 2: ***Associate Graph***
Associate graph is a graph which describes the associate relationship between each pair of items. It can be formulized as:

$$G_{graph} = \mathbf{Graph}(Nodes, Edges),$$

in which each node represents a resource item; each edge represents the associate relationship between two linked nodes, and the weight refers to the firmness of the associate relationship.

Figure 4 is an example of associate graph.

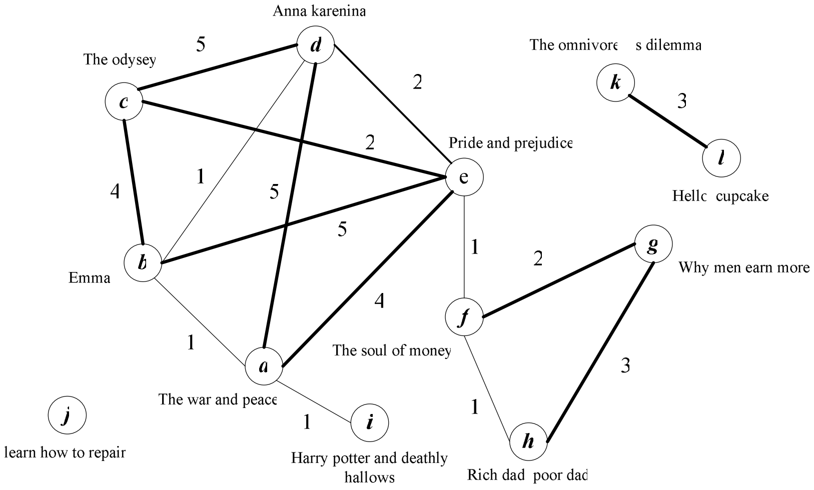


Figure 4: An Example of Associate Graph

5.2 Creation of Meta Data User Interest Modelling

The user modeling constructed here is the same as the one in domain category tree. Usage behavior is used to extract a list of the user‘s favored items, as in the case of user interest model discussed above:

$$u_i \equiv favList = \{i_1, i_2, \cdots, i_n\} = \{(i_1, v_1), (i_2, v_2), \cdots, (i_n, v_n)\},$$

each pair within the parenthesis represents a favored item, in which *i* is the item name and *v* is the corresponding favor value.

5.3 Creation of Meta Data Associate Graph Matching

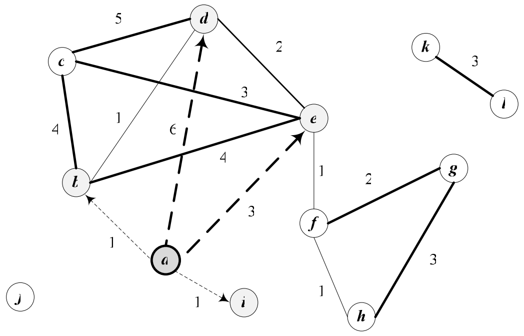
We design a novel graph propagation method to conduct our associate graph matching. For each pair $(i_\square, v_\square)$ in the list of favored items, we find out the corresponding node in the associate graph, and then propagate the item value *v* through weighted edges layer by layer hierarchically, with the whole process controlled by certain stop conditions. After each item pair in the list of favored items has been processed, we can integrate all these included nodes together to get the first round graph matching nodes set. Next we propagate nodes with highest scores in the set and update the set iteratively until certain stop conditions are met. Then the final graph match sore of each node, S_{graph} , is obtained. The detailed process is illustrated in Algorithm 2.

Algorithm 2: *Graph Propagation*

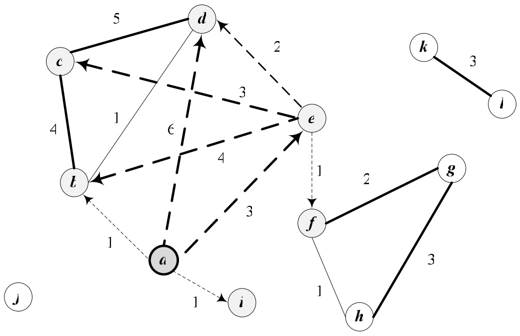
```
Input: $I_{graph} = Graph(nodes, edges)$, an associate graph
       $u \equiv favList = \{(i_1, v_1), (i_2, v_2), \cdots, (i_n, v_n)\}$, a user favor model
       $M_e = \begin{pmatrix} e_{11} & \cdots & e_{1n} \\ \vdots & \ddots & \vdots \\ e_{n1} & \cdots & e_{nn} \end{pmatrix}$, edge matrix of the graph
Output: $S_{graph} = (s_1, s_2, \cdots, s_n)$, the matching score of each node in graph
Define:
 begin
  //match stage
   recList = {};
  for each $node_k$ in the associate graph $I_{graph}$ do
    begin
      $i_k = node_k \to item_k$ ;    //get the according item of $node_k$
      Search $i_k$ in the favList ;
      if $i_k$ in the favList and corresponding pair is $(i_k, v_k)$ then
        $s_k := v_k$ ;
        Put $node_k$ into $recList$ ;
      else
        $s_k := 0$ ;
     end for.
   //propagation stage
   while Count (recList) < threshold then do
     begin
        Select node with highest score $node_k$ in recList
        Propagate $node_k$ to its neighbor nodes $[\{node\}_i]_{i=1:m}$
        Put $[\{node\}_i]_{i=1:m}$ into $recList$
        for each node $node_j$ in recList do
          begin
             Re-AssignScore($node_j$);    //re-assign all nodes's score in recList
          end for.
       end while.
   end.
```

Traditional associate method could be seen as a simple graph propagation with only one-layer propagation. In the example shown in Figure 5(a): if item *a* is favored by the user, then items *b*, *d*, *e*, *i* would be associate items, so that just by ranking them according to their associate weights, we can get the associate recommendation list $l_{associate} = \{d, e, i, b\}$; and in the same example shown in Figure 5(b), item *a* would propagate its favored value to its neighbor items in the first round, *b*, *d*, *e*, *i*, just the same as the above, then we choose item *c*, the one with the highest edge weight sum in the current sub-graph, as the seed item to conduct the second round propagation; here two rounds of propagation are enough and we rank all selected items by their edge weight sum in the current sub-graph to get the final recommendation list $l_{propagation} = \{e, d, b, c, i\}$. It is worth noting that, we choose an item's edge weight sum in the current sub-graph as ranking criterion because it not only indicates the popularity of this item, but also reflects the tightness of its relationship with original favored items. Comparing the two result lists $l_{associate}$ and $l_{propagation}$, we see that our graph propagation method has two advantages:

1) Ability to discover missed correlative items. In our example, item *c*, which has strong relation with item *a*, is missed in case the traditional associate method is used. Our proposed method could help solve this problem effectively.
2) Accuracy in ranking the items in result list. Look at item *b* and item *i* in the above example, with the traditional associate method, there is no difference between them, so that they are hard to be ranked correctly, but in the graph propagation method, item *b* is clearly more related to item a than item *i* to that, in exact accordance with the actual situation.



(a) Traditional Associate Graph



(b) Graph Propagations

Figure 5: Algorithm Analysis

6 Experimental Evaluation

We have conducted simulation experiments and user study to evaluate our proposed recommendation method, which are discussed in this section. First, we introduce the experiment dataset, Book-Crossing (BX) dataset. Then we describe the simulation experiments and analyze the results. User study is elaborated at last to show the practical performance of our method.

6.1 Data

We choose Book-Crossing [Ziegler et al., 2005] dataset as the experiment dataset. Book-Crossing dataset was collected in a 4-week crawl (August/September 2004) from the Book-Crossing community. It contains the data on 278,858 users providing 1,149,780 ratings about 271,379 books.

To simulate the non-register situation, we treat the book with a rating higher than 5 as this user's favorite book (the rating values range from 0 to 10). Then each user has a favorite book list, which is simulatively treated as a click-through chain. The click-through chain sets are randomly divided into two sets: main set and test set. The main set is used to construct associate inverted index and associate graph, and the test set to conduct evaluation, in which each click-through chain is divided into existing purchases and potential purchases.

6.2 Simulation

To evaluate the performance of different recommendation methods, we adopted a holdout testing approach [Huang et al., 2004]. For each target user, we divide her/his click-through chain *l* into two parts, l_{head} and l_{tail} . The first part can be treated as existing purchase list, fuctioning as input to be fed into different methods to generate the recommendation list. For comparison purpose, the second part can be treated as potential purchase list of the user and it is invisible to the recommender system. The detailed process is shown in Algorithm 3.

Algorithm 3: *Holdout Testing*

```
Input: $S_{chain} = \{c_1, c_2, \cdots, c_n\}$, a click-through chain set used for evaluation
       Recommender, a recommendation system to be evaluation
Output: evaluation value $p$
Define:
  begin
    for each click chain $c_i$ in the $S_{chain}$ do
      begin
        Randomly divide $c_i$ into two parts, $c_{head}$ and $c_{tail}$ ;
        Treat $c_{head}$ as certain user favor item list
            $l_{fav}$ : = $c_{head}$ ;
            $l_{rec}$ := Recommender ($l_{fav}$) ;     //get the recommending item list
        Evaluate by compare $l_{rec}$ and $c_{tail}$
        $p_i$ := computePrecition($l_{rec}$ , $c_{tail}$ ) ;
      end for.
      $p$ := average($p_i$ , i=1:n) ;
  end.
```

First of all, we evaluate our proposed associate graph propagation method. Three methods are compared in our simulation experiment: category tree matching, traditional associate method and graph propagation.

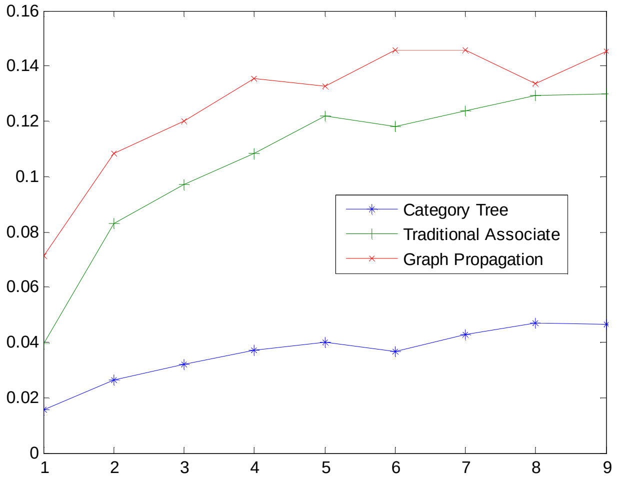


Figure 6: Associate Graph Recommendation Result Chart

In Figure 6, the x-coordinate shows the length of inputted existing purchase list, and the y-coordinate shows the recall value, a classical evaluation criterion in the information retrieval area. We can discern four facts from the chart:

1) The recall of the category tree matching is low compared with that of the other two methods. The reason is that in our experiment dataset, the only available attributes are authors and publishers of the books, which are relatively insufficient for the category tree construction.
2) The overall trends of these three curves show that the recall value increases along with the length of existing purchase list. With the intuition, the more items are provided in the existing purchase list, the more user interest information is collected, with the recommended list being more accurate. Hence the conclusion that our experiment result is in accordance with the actual situation.
3) Comparing the two curves of associate methods, we see that the graph propagation curve reaches its peak value and maintains a relatively steady value quicker than the traditional associate method. The recall value of graph propagation could reach an approximate peak value when the length of inputted favorite list is four, which means the system only need to collect about four favorite items for certain user to provide a satisfying recommendation service. This means the cold start problem could be solved with our method to a certain extent, which is especially meaningful in the non-register environment where only limited user information can be collected in a short session time.
4) It is obvious that our proposed graph propagation method outperforms the traditional associate recommendation by nearly 20%. The chart also shows that the superiority of our method over traditional associate is more obvious when the length of inputted purchase list is relatively short. This in a way reflects the characteristics of both methods: the traditional associate recommendation method can actually be seen as a simple associate graph propagation with only one-layer propagation. Therefore, when the inputted purchase list is relatively short, our proposed method with hierarchical propagation could dig out more related items than the traditional method. When the length of existing purchase list increases, the traditional method with one-layer propagation may be enough to dig out sufficient related items, and thus the gap between two methods tends to be smaller with the increase of the length of purchases list.

We select the recall value as our evaluation criterion for the reason that the length of the potential purchase list is dynamically changed in this scenario and the recall value could normalize the length difference better compared with the precision value. Another point worth particular notice is that the result value is relatively small because the potential purchase list is too short to cover all the books in which the user is interested. This limitation of our similation evaluation can be complemented by user study.

Secondly, we evaluate our joint recommendation method which combines together the domain category tree matching and associate graph matching. Here we need to conduct an experiment to select appropriate joint weight first, specifically, the proportion each method takes in our new joint recommendation method. In this experiment, we tune the joint weight of domain category tree matching from 0 to 1 with the step length set as 0.1, by comparing the performance measured by the accumulative recall.

Figure 7 illustrates the weight training result. From this figure we see that 0.1 is the best joint weight, which means a relatively smaller proportion of the category tree matching is preferred in our scenario. The reason is that our Book-Crossing only contains authors and publishers as useful attributes when the domain category tree is organized, while on the other hand, the click-through chain data for associate graph are relatively adequate. Therefore 0.1 is chosen as the joint weight of our joint recommendation method.

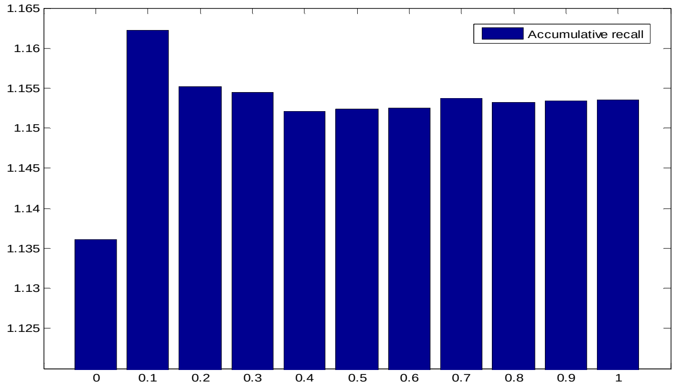


Figure 7: Joint Weight Adjustment

Figure 8 illustrates the effect of our newly proposed joint method compared with that of the single graph associate method. It shows that the joint method does outperform the single graph propagation method (it says nothing of the single domain category tree method). We may easily notice that the improvement is relatively small, for which there are two reasons: 1) the domain category tree is relatively weak here because only two useful attributes are available during its organization 2) the length of potential purchase list is too short to cover all books in which the user is interested.

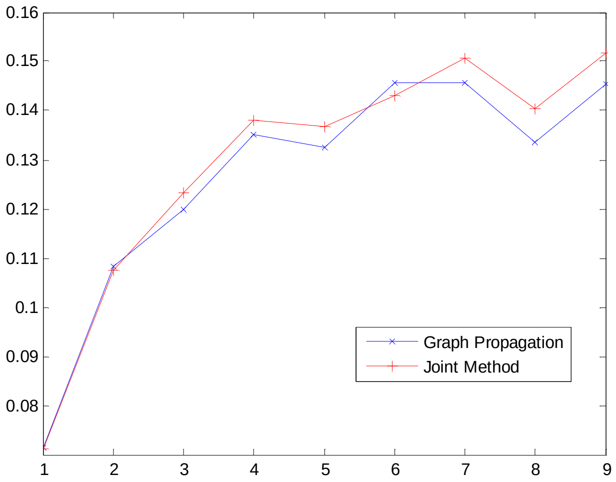


Figure 8: Comparison of the Result of Graph Propagation and Joint Method

6.3 User Study

In simulation, we can conduct large scale tests to train parameters and roughly compare the results of different methods, but it is still hard to get a clear picture of the practical effect of the system. User study proves an effective way to make up such deficiency. In such study, we ask a group of users to experience our systems, with three different methods provided to each user: the traditional associate recommendation, which is treated as a baseline; the associate graph propagation method, which is taken to evaluate our novel idea associate graph; and the joint method combining associate graph matching and category tree matching, which is taken to evaluate the joint method. Each tested user is asked to search books in the system and mark their favorite books. When enough books are marked by a certain user, say, five books, each method will recommend an book list to this user, so that altogether three lists would be recommended to each user. Then the user would label whether the books in each recommended list are those she/he is interested or not.

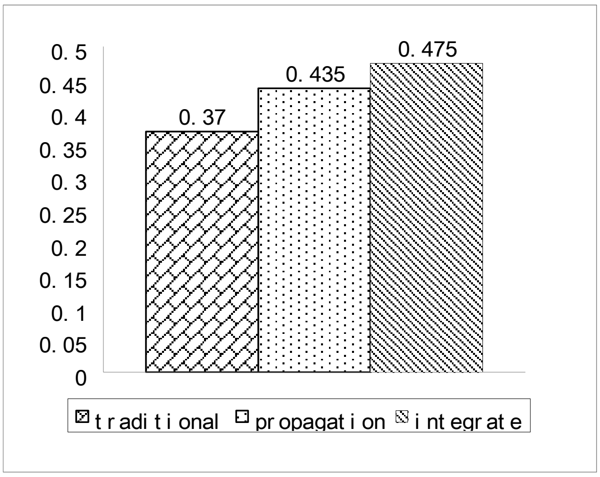


Figure 9: User Study Result

Figure 9 shows the result of the user study described above, in which we see that our joint recommendation method and the new graph propagation recommendation indeed improve performance of recommendation services. One thing that deserves our notice is that the improvement of joint method is relatively remarkable compared with the one in the simulation evaluation. In simulation, the potential purchase list obtained from the user's purchase record is too short to cover all the books the user is interested in, so that most of such books may be missed in the potential purchase list. This deficiency could be complemented by user study, since the user may add her/his choices with subjective judgement, so the remarkable improvement of the joint method in user study can be an indication of its practical effectiveness.

7 Conclusion and Future Work

In this paper, we have proposed a novel joint recommendation method based on the domain category tree and associate graph, which can provide satisfactory recommendation services in non-register environments. The limited data available in the non-register environment is fully utilized: the domain category tree was built up with the content and attributes of the items to make full use of the content feature, and the associate graph was constructed with click-through chains of user sessions to tap the full potential of the mass associate feature. We have evaluated our method with a case where books are recommended, and experimental results show that the joint method can provide better recommendation services than either category tree matching or associate filtering used alone, and our novel graph propagation method can indeed outperform the traditional associate recommendation in the non-register environment.

Surely there is still much work needs to be done in this specific field. Firstly, it is important to design a more effective and efficient associate graph matching method, which is the key innovation of our method. Secondly, apart from traditional recommendation applications, we can also apply our method to personalized searching by combining together the primary query score and our recommending score to re-rank the search results. Weight tuning would be the main task in this new application. Finally, we plan to apply our method in the practical courseware recommendation platform.

References

[Adomavicius and Tuzhilin, 2005] Adomavicius, G., and Tuzhilin, A.: “Toward the Next Generation of Recommender Systems: a Survey of the State-of-the-art and Possible Extensions”; *IEEE Transactions on Knowledge and Data Engineering*, 2005(June), 17(6): 734-749.

[Basilico and Hofmann, 2004] Basilico, J., and Hofmann, T.: “A Joint Framework for Collaborative and Content Filtering”; *Proceedings of the Twenty-seventh Annual International SIGIR Conference on Research and Development in Information Retrieval* (SIGIR’04), Sheffield, 2004, pp. 550-551.

[Blei et al., 2003] Blei, D., Ng, A., and Jordan, M.: “Latent Dirichlet Allocation”; *Journal of Machine Learning Research*, 2003(Jan), 3: 993–1022.

[Bollacker et al., 2000] Bollacker, K.D., Lawrence, S., and Giles, C.L.: “Discovering Relevant Scientific Literature on the Web”; *IEEE Intelligent Systems*, 2000, 15(2): 42-47.

[Brin et al., 1997] Brin, S., Motowani, R., and Silverstein, C.: “Beyond Market Basket: Generalizing Association Rules to Correlations”. *Proceedings of the ACM SIGMOD Conference on Management of Data* (SIGMOD’97). New York, 1997, ACM Press: 265-276.

[Burges et al., 2005] Burges, C.J.C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., and Hullender, G.: “Learning to Rank using Gradient Descent”; *Proceedings of the 22nd International Conference on Machine Learning*, Bonn, 2005.

[Chen et al., 2001] Chen, C.C., Chen, M.C., and Sun, Y.S.: “PVA: a Self-Adaptive Personal View Agent System”; *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Schkolnick M., Provost F., Srikant R. (eds.), New York, 2001, ACM Press, pp. 257~262.

[Claypool et al., 1999] Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., and Saitin, M.: “Combining Content-Based and Collaborative Filters in an Online Newspaper”; *Proceedings of ACM SIGIR’99 Workshop Recommender Systems: Algorithms and Evaluation*, Aug, 1999.

[Cortes and Vapnik, 1995] Cortes, C., and Vapnik, V.: “Support Vector Networks”; *Machine Learning*, 20, 1995.

[Domingos and Pazzani, 1997] Domingos, P., and Pazzani, M.: “On the Optimality of the Simple Bayesian Classifier under Zero-One Loss”; *Machine Learning*, 1997, 29:103-137.

[Girard, 2008] Girard, J.: “Diapers, Pop-Tarts, and Dog Food”; http://www.prairiebizmag.com.

[Goldberg et al., 1992] Goldberg, D., Nichols, D., Oki, B.M., and Terry, D.: “Using collaborative filtering to weave an information tapestry”; *Communications of the ACM*, 1992, 35 (12): 61–70.

[Hofmann, 1999] Hofmann, T.: “Probabilistic Latent Semantic Indexing”; *Proceedings of the Twenty-Second Annual International SIGIR Conference on Research and Development in Information Retrieval* (SIGIR’99), California, 1999.

[Huang et al., 2002] Huang, Z., Chung, W.Y., Ong, T.H., and Chen, H.C.: “A Graph-Based Recommendation System for Digital Library”; *Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital Libraries* (JCDL’02), Portland, Oregon, USA.

[Huang et al., 2004] Huang, Z., Chen, H., and Zeng, D.: “Applying Associative Retrieval Techniques to Alleviate the Sparsity Problem in Collaborative Filtering”; *ACM Transactions on Information Systems* (TOIS’04), 22(1): 116-142, January

[Joachims et al., 1997] Joachims, T., Freitag, D., and Mitchell, T.: “WebWatcher: A Tour Guide for the World Wide Web”; *Proceedings of the International Joint Conference on Artificial Intelligence*. Georgeff, M.P., Pollack, E.M. (eds.), San Francisco: Morgan Kaufmann Publishers, 1997, 770~777.

[Jolliffe, 2002] Jolliffe, I.T.: “Principal Component Analysis”; Series: Springer Series in Statistics, 2nd ed., Springer, NY, 2002, XXIX, 487 p. 28 illus. ISBN 978-0-387-95442-4.

[Koren et al., 2008] Koren, J., Zhang, Y., and Liu, X.: “Personalized interactive faceted search”; *Proceedings of the 17th International Conference on World Wide Web* (WWW’08), Beijing, China.

[Linden et al., 2003] Linden, G., Smith, B., and York, J.: “Amazon.com Recommendations: Item-to-Item Collaborative Filtering”; *Internet Computing*, 2003, 7:76-80, IEEE.

[Melville et al., 2002] Melville, P., Mooney, R.J., and Nagarajan, R.: “Content-Boosted Collaborative Filtering for Improved Recommendations”; *Proceedings of the eighteenth Nat’l Conf. Artificial Intelligence*, 2002.

[Mladenic, 2000] Mladenic, D.: “Machine Learning for Better Web Browsing”; *AAAI 2000 Spring Symposium Technical Reports on Adaptive User Interfaces*. Rogers S., Iba W. (eds.), Menlo Park, CA: AAAI Press, 2000, 82~84.

[Mostafa et al., 1997] Mostafa, J., Lam, S.W., and Palakal, M.: “A Multilevel Approach to Intelligent Information Filtering: Model, System, and Evaluation”; *ACM Transactions on Information Systems*, 1997, 15(4):368-399.

[Pazzani et al., 1996] Pazzani, M., Muramatsu, J., and Billsus, D.: “Syskill & Webert: Identifying Interesting Web Sites”; *Proceedings of the Thirteenth National Conference on Artificial Intelligence* (AAAI 96), Portland, Oregon, pp. 54-61.

[Pazzani, 1999] Pazzani, M.: “A Framework for Collaborative, Content-Based, and Demographic Filtering”; *Artificial Intelligence Rev.*, pp. 393-408, Dec. 1999.

[Pedro and Pazzani, 1997] Pedro, D., and Pazzani, M.: “On the Optimality of the Simple Bayesian Classifier under Zero-One Loss”; *Machine Learning*, 1997, 29:103–137.

[Pohl et al., 2007] Pohl S., Radlinski F. and Joachims T.: “Recommending Related Papers based on Digital Library Access Records”; *The Joint Conference on Digital Libraries* (JCDL’07), June 18-23, Vancouver, British Columbia, Canada.

[Resnick et al., 1994] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J.: “GroupLens: An Open Architecture for Collaborative Filtering of Netnews”; P*roceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, Chapel Hill, NC, 1994, pp. 175-186.

[Sarwar et al., 2001] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J.: **“**Item-Based Collaborative Filtering Recommendation Algorithms”; *Proceedings of the 10th International World Wide Web Conference* (WWW’01), May 1-5, 2001, Hong Kong. ACM 1-58113-348-0/01/0005.

[Shakhnarovich et al., 2005] Shakhnarovich, G., Darrell, T., and Indyk, P.: “Nearest-Neighbor Methods in Learning and Vision”; *The MIT Press*, 2005, ISBN 0-262-19547-X.

[Shardanand and Maes, 1995] Shardanand, U., and Maes, P.: “Social Information Filtering: Algorithms for Automating Word of Mouth”; *Proceedings of the ACM CHI’95 Conference on Human Factors in Computing Systems*. Roberts T., Robertson S. (eds.), New York, 1995: ACM Press, 210~217.

[Srivastava et al., 2000] Srivastava, J., Cooley, R., and Deshpande, M.: “Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data”; *Proceedings of the ACM SIGKDD Explorations*. New York, 2000, ACM Press, 1(2):12-23.

[Tunkelang, 2006] Tunkelang, D.: “Dynamic Category Sets: An approach for Faceted Search”; *Proceedings of the ACM SIGIR’06 Workshop on Faceted Search*, Aug. 2006.

[Zhang and Fang, 2005] Zhang, Y.X., and Fang, C.H.: “Active Service: Concept, Architecture and Implementation”; *Thomson Learning*, USA, 2005.

[Zhou et al., 2005] Zhou, D., Schölkopf, B., and Hofmann, T.: “Semi-Supervised Learning on Directed Graphs”; *Advances in Neural Information Processing Systems* 17, 1633-1640. L.K. Saul, Y. Weiss and L. Bottou (Eds.), MIT Press, Cambridge, MA, 2005.

[Zhou et al., 2005] Zhou, D., Huang, J., and Schölkopf, B.: “Learning from Labeled and Unlabeled Data on a Directed Graph”; *Proceedings of the 22nd International Conference on Machine Learning*(ICML’05), 1041-1048. (Eds.) L. De Raedt and S. Wrobel, ACM press, 2005.

[Ziegler et al., 2005] Ziegler, C., McNee, S.M., Konstan, J.A., and Lausen, G.: “Improving Recommendation Lists through Topic Diversification”; *Proceedings of the 14th International World Wide Web Conference* (WWW '05), May 10-14, Chiba, Japan.

[Zhu and Ghah, 2002] Zhu, X., and Ghahramani, Z.: “Learning from Labeled and Unlabeled Data with Label Propagation”; *Technical Report* CMU-CALD-02-107, Carnegie Mellon University.