WEB INFORMATION RECOMMENDATION MAKING BASED ON ITEM TAXONOMY

LI-Tung Weng, Yue Xu, Yuefeng Li and Richi Nayak
Faculty of Information Technology, Queensland University of Technology, 4001 Queensland, Australia

Keywords: Recommender System, Taxonomy, Ecommerce, Cold-start Problem.

Abstract: Recommender systems have been widely applied in the domain of ecommerce. They have caught much research attention in recent years. They make recommendations to users by exploiting past users' item preferences, thus eliminating the needs for users to form their queries explicitly. However, recommender systems' performance can be easily affected when there are no sufficient item preferences data provided by previous users. This problem is commonly referred to as cold-start problem. This paper suggests another information source, item taxonomies, in addition to item preferences for assisting recommendation making. Item taxonomy information has been popularly applied in diverse ecommerce domains for product or content classification, and therefore can be easily obtained and adapted by recommender systems. In this paper, we investigate the implicit relations between users' item preferences and taxonomic preferences, suggest and verify using information gain that users who share similar item preferences may also share similar taxonomic preferences. Under this assumption, a novel recommendation technique is proposed that combines the users' item preferences and the additional taxonomic preferences together to make better quality recommendations as well as alleviate the cold-start problem. Empirical evaluations to this approach are conducted and the results show that the proposed technique outperforms other existing techniques in both recommendation quality and computation efficiency.

1 INTRODUCTION

Recommender systems have been an active research area for more than a decade, and many different techniques and systems with distinct strengths have been developed (Montaner et al. 2003). Among all these different recommendation techniques, collaborative filtering is perhaps the most successful and widely applied technique for building recommender systems(Deshpande and Karypis 2004; Schafer et al. 2000). In general, collaborative filtering based recommenders recommend items that are commonly preferred by users with similar item preferences to a target user. Therefore, the recommendation quality of the collaborative filtering technique depends upon the number of users with similar preferences to the target user. If there are only few users in the dataset with similar preferences to the target user, then the standard collaborative filtering technique will not be able to suggest quality recommendation to the user. This issue, commonly referred to as cold-start problem(Schein et al. 2002), usually happens when the system is newly built (there is no initial data in the dataset), or when there is no data available for a new target user(Middleton et al. 2002).

A commonly used approach to alleviate the cold-start problem is to take item content information into consideration in recommendation making. That is, when it is not possible to form a neighbourhood for a target user, content based techniques can be used to mine the item contents preferred by the target user, and based on the preferred item contents the recommendations can be generated by finding items with similar contents preferred by the target user (Burke 2002; Sarwar et al. 2000). However, because most of the content based techniques represent item content information as word vectors and maintain no semantic relations among the words, therefore the result recommendations are usually very content centric and poor in quality(Adomavicius et al. 2005; Burke 2002; Ferman et al. 2002; Sarwar et al. 2000). To improve the content based techniques, the content information for the items should be captured in more sophisticated ways so that associations among items can be measured by their content semantic meanings rather than simple keywords mappings.

Weng L., Xu Y., Li Y. and Nayak R. (2008).
WEB INFORMATION RECOMMENDATION MAKING BASED ON ITEM TAXONOMY.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - SAIC*, pages 20-28
DOI: 10.5220/0001695100200028

In this paper, we propose a novel recommendation making approach, namely Hybrid Taxonomy based Recommender (HTR), which generates item recommendations based on both users' item preferences and item taxonomic preferences. The notion of item taxonomy information is used in our system in place of standard item content information, that is, instead of using keywords vectors to represent items, our system describes items based on taxonomic topics extracted from a tree-like taxonomy structure. The item taxonomy information is useful for encapsulating item content semantics as it allows items with different topics to be related if they share common supper topics. Hence, not only the use of item taxonomy can significantly alleviate the cold-start problem, but it can also improve recommendation quality by reducing the content centric issue. The relationship between the item preferences and the item taxonomic preferences is also investigated in this paper. Based on our study and experiments, we suggest that when a set of users shares similar item preferences, they might also share similar item taxonomic preferences. The HTR technique utilizes the proposed relation to achieve competitive computation efficiency and recommendation performance. For the applicability concern, as item taxonomy information is available for most e-commerce sites and standardization organizations, HTR can be easily applied and adopted to a wide range of domains. Moreover, HTR can also adopt the implicit user preference information (in addition to the standard explicit user preferences) to further enhance its recommendation quality in cold-start environments.

2 RELATED WORK

Much research has suggested that the cold-start problem can be alleviated by combining collaborative filtering and content based techniques together (Burke 2002; Ferman et al. 2002; Park et al. 2006; Schein et al. 2002). However, because part of the recommendation process for these hybrid recommenders is content-based, the generated recommendations may be excessively content centric and lack of novelty(Middleton et al. 2002; Ziegler et al. 2004). Hence, semantic and ontology based techniques have been suggested to improve the recommendation generality for the content based filtering. Middleton(Middleton et al. 2002) suggested an ontology based recommender which uses external organizational ontology (e.g. publication and authorship relationships, projects and project membership relationships, etc.) to solve the cold start problem. However, as the Middleton's technique is mainly designed for recommending research papers and documents, and also relies on a specific organizational ontology, therefore it is not easy to adopt this method for general recommenders. On the other hand, Ziegler(Ziegler et al. 2004) proposed a taxonomy-driven product recommender (TPR), it utilizes a general tree structured product taxonomy to enhance its recommendations. Due to the simplicity of the taxonomy structure, Ziegler's technique is considered widely applicable to different domains(Ziegler et al. 2004). To the best of our knowledge, Middleton and Ziegler's techniques are the only two works bearing traits similar to the proposed HTR technique. HTR employs similar tree structured taxonomy to TPR, and therefore it inherits TPR's generality advantage. However, while TPR only considers implicit item preferences for making recommendations, HTR utilizes the relationship between users' explicit item preference and implicit taxonomic preferences for recommendation making, therefore yields better recommendation performances. Moreover, HTR adopts item-based collaborative filtering paradigm (Deshpande and Karypis 2004) in contrast to TPR's user-based collaborative filtering. Item-based collaborative filtering allows most computations to be done offline. Therefore, the computation efficiency of online recommendation generation can be improved.

3 PROPOSED APPROACH

The idea behind HTR is intuitive. It firstly finds a set of users with similar preferences to a given target user, and then extracts taxonomy topics that are popularly and uniquely preferred by these users. Finally, HTR estimate the target user's preference to a candidate item by combining user item preferences with taxonomy topic preferences.

This section is divided into five parts. In Section 3.1, the basic system model and general notations used throughout this paper are described. In Section 3.2, we discuss the implicit relation between users' item preferences and taxonomic preferences. The technique for taxonomic preference extraction is described in Section 3.3. At last, Section 3.4 details the proposed HTR method.

3.1 System Model

We envision a world with a set of users

$U = \{u_1, u_2, \dots, u_n\}$ and a set of items $T = \{t_1, t_2, \dots, t_m\}$. Each user $u \in U$ is associated with a set of rated items $RT_{all}(u) \subseteq T$. Based on the different rating methods, we can divide these items into implicitly rated items $RT_{impl}(u) \subseteq RT_{all}(u)$ and explicitly rated items $RT_{expl}(u) \subseteq RT_{all}(u)$. A user can rate an item implicitly or explicitly, but not both (i.e.$RT_{expl}(u) \cap RT_{impl}(u) = \emptyset$).

In explicit ratings, users express their preferences to items in numeric form, that is, the value 0 indicates minimal satisfaction and 1 indicates maximum satisfaction. We use $rating(u,t)$ to denote user $u \in U$'s rating to item $t \in RT_{expl}(u)$, such that $0 \leq rating(u,t) \leq 1$.

HTR uses taxonomy based descriptors to describe items. Specifically, $D(t) = \{d_1, d_2, \dots, d_n\}$ denotes a set of descriptors characterizing any item $t \in T$'s taxonomy, where $|D(t)| \geq 1$. A taxonomy descriptor is a sequence of ordered taxonomic topics, denoted by $d = (p_0, p_1, \dots, p_q)$ where $d \in D(t)$, $t \in T$. The topics within a descriptor are sequenced so that the former topics are super topics of the latter topics, specifically, p_j is the direct super topic for p_{j+1} where $0 \leq j < q$. A super topic covers a broader concept than its sub-topics, and a topic can have more than one direct sub-topics. Thus, it is easy to envision that the taxonomy topics are stored in a tree-like structure, and the tree structure formed with the taxonomy topics is referred as the taxonomy tree, and all item descriptors are paths that are extracted from the root to a leaf node on the tree.

Let C be the set of all taxonomy topics, $C = \{p | p \in d, d \in D(t), t \in T\}$ and $E: C \rightarrow 2^C$ be a map from C to 2^C that retrieves all direct sub-topics $E(p_a) \subset C$ for topics $p_a \in C$. Based on E, we define a partial order $\prec$ on the taxonomy topic set C to differentiate between super topics and sub-topics. $\forall p_a, p_b \in C$, if $p_a \in E(p_b)$, then p_a and p_b have the relationship $\prec$, i.e., $p_a \prec p_b$ require that $E(p_a) \cap E(p_b) = \emptyset$ for all $p_a, p_b \in C$, $a \neq b$. With this requirement and the map E, we can recursively extract the taxonomy tree structure from the set C. Moreover, as in standard tree structures, the taxonomy tree has exactly one top-most element with zero in-degree representing the most general topic, it is denoted by Γ in this paper. By contrast, for these bottom-most elements with zero out-degree, they are denoted by $\perp$ and represent the most specific topics. In our system, for any item descriptor $d = (p_0, p_1, \dots, p_q)$, it is required $p_0 = \Gamma$ and $p_q = \perp$.

3.2 Cluster-based User Neighbourhood

In HTR, cluster based neighbourhood formation is adopted to ensure the computation efficiency. In order to form the user neighbourhoods or clusters, a similarity measure for computing user similarities is essential. In HTR, we adopted the correlation measure described in (Breese et al. 1998) to compute the item preference similarity between two users $u_i, u_j \in U$ as given in Equation (1).

$$sim(u_i, u_j) = \frac{\sum_t (rating(u_i,t) - avg(u_i))(rating(u_j,t) - avg(u_j))}{\sqrt{\sum_t (rating(u_i,t) - avg(u_i))^2 (rating(u_j,t) - avg(u_j))^2}} \quad (1)$$

where t is an item rated explicitly by both u_i and u_j, that is $t \in RT_{expl}(u_i) \cap RT_{expl}(u_j)$. $avg(u)$ denotes the average explicit ratings made by $u \in U$.

Based on Equation (1), U can be divided into a set of clusters $UC = \{uc_1, uc_2, \dots, uc_k\}$, such that $\bigcup_{uc \in UC} uc = U$ and $\bigcap_{uc \in UC} uc = \emptyset$. For the sake of convenience, let $cluster(u) \in UC$ denote the cluster which contains user u. Because the clusters are constructed based on users' item preference similarity, users within the same cluster will have similar item preferences. In this paper, we take a further investigation to suggest the following assumption:

users within the same neighbourhood or cluster sharing similar item preferences may share similar taxonomic preferences and interests

The idea behind the assumption is that the users within one cluster should have apparent similar taxonomic focus and the taxonomic focuses of the users in different clusters should be different. In this paper, we use information gain to measure the certainty of taxonomy focus of a user set, and empirically demonstrate the validity of the above assumption by using information gain measure. When the information gain is high, it indicates that the certainty of the taxonomic focuses of user clusters is high. Therefore we can use information gain to investigate whether different clusters have apparent taxonomic focuses and the taxonomic focuses are different in different user clusters. The adapted information gain can be calculated as below:

$$Gain = H(U) - \sum_{uc \in UC} \Pr(uc) \times H(uc) \quad (2)$$

where $\Pr(uc)$ is the probability that an item rating is made by a user in cluster uc. $H(U)$ is the information entropy for a given user space. The concept of information entropy is adapted in this paper to measure the degree of taxonomic focus in a user set (i.e. a cluster or a neighbourhood). If the information entropy is high for a user set, then there is no apparent taxonomic focuses in the set (i.e. users in the set prefer all taxonomy topics equally), and vice versa. The information entropy formula is depicted below:

$$H(U') = \sum_{p \in C, p=\perp} -\Pr(p, U') \times log_2 \Pr(p, U') \quad (3)$$

In the entropy equation, $\Pr(\mathrm{p}, \mathrm{U}')$ denotes the probability that the users in the user set $\mathrm{U}' \subseteq \mathrm{U}$ are interested in the taxonomy topic p. For a given clustering $\mathrm{UC} = \{\mathrm{uc}_1, \mathrm{uc}_2, \ldots, \mathrm{uc}_\mathrm{k}\}$, if $\mathrm{H}(uc_\mathrm{i})$ are low which means the taxonomic focuses are apparent in cluster uc, according to Equation (2), the information gain is high.

The effect of user clustering on taxonomy information gain is depicted in Table 1. This result is obtained by using k-means clustering technique to divide 278,858 users in "Book-Crossing" dataset (www.informatik.uni-freiburg.de/~cziegler/BX/) into 100 clusters according to their explicit ratings. We have tried to produce different number of clusters for the dataset (i.e. different values for k), and we have found by setting k to 100 (i.e. 100 clusters) can produce clusters with reasonable qualities.

Table 1: The effect of user clustering on taxonomy information gain.

	Explicit Ratings	Explicit + Implicit Ratings
users clusters formed based on user ratings	0.823	0.458
Randomly formed user clusters (baseline)	-0.385	-0.319

Our first experiment is to show if user clusters have stronger taxonomic focuses than the entire dataset when only explicit ratings are considered. It is shown in the first column of Table 1, the result information gain is 0.823, which is a big increase when comparing it with the information gain obtained from the randomly formed cluster partitions (i.e. -0.385). This result shows that, by clustering users with their explicit ratings, each user cluster has its own taxonomic focuses.

Because our clusters are generated based on only explicit ratings, it might be unfair if we only consider explicit ratings in calculating taxonomy information gain. Hence, we further include the implicit ratings in computing taxonomy information gain. With identical cluster settings, we still get a strong information gain increase (i.e. 0.458) when comparing to the information gain obtained from the random formed clusters (i.e. -0.319). Based on the information gain analysis, we can conclude that users within the same clusters not only share similar item preferences, but they also share similar taxonomic preferences.

3.3 Taxonomic Preferences Extraction

For each cluster $uc \in UC$, we build a cluster based taxonomy tree similar to the global taxonomy tree defined in Section 3.1. Formally, we define the cluster based topic set:

$$C_{uc} = \{p | p \in d, d \in D(t), t \in RT_{all}(u), u \in uc\}$$

and $E_{uc}(p) \subset C_{uc}$ for topics $p \in C_{uc}$ extracts the direct sub-topics of p.

Using the similar way described in Section 3.1, with the map E_{uc}, we can construct a local taxonomy tree from a cluster uc. With the local cluster based taxonomy tree, we can then find the frequent and distinct topics for each cluster. We measure the distinctness of a topic p within a local cluster uc in accordance to the global user set by:

$$topic_{score(p,C_{uc})} = \begin{cases} 0, & t\_count(p, uc) < \psi \\ \frac{t\_count(p,uc)}{t\_count(p,U)}, & otherwise \end{cases} \quad (4)$$

where $t\_count(p, U')$ is the number of user ratings to items involving taxonomy topic p within a given user set $U' \subseteq U$. ψ is a user defined constant, it is used to filter out topics that are not popularly interested by users. In this paper, ψ is set to 50. So topics need to be involved in at least 50 ratings in order to get a reasonable score.

The higher the topic score, the higher the possibility the taxonomy topic is unique to a cluster. Based on the topic score, the topics with their topic scores higher than a predefined threshold are chosen as the hot topics for that cluster. We denote the hot topic set by:

$$hot\_topics(uc, \zeta) = \{p | p \in C_{uc}, topic\_score(p, C_{uc}) > \zeta\} \quad (5)$$

where ζ is the user defined threshold. In our experiment, ζ is set to 0.6. Figure 1 shows the average number of topics left for each cluster for different threshold settings.

For the "Book-Crossing" dataset there are originally 10746 topics in the entire dataset. After user clustering, the average number of topics per cluster is around 3164.12. The ratio of the topic number in the clusters out of the topic number in the entire dataset is about 0.29. This ratio suggests that different clusters may have very different taxonomy topics. Moreover, after we increase the topic score threshold ζ, the ratio decreases drastically (e.g. when $\zeta = 0.68$, the entire dataset has 530 topics and the average number of topics per cluster is 5.9, the ratio is only 0.01.). This observation further strengthens the conclusion that we made about cluster taxonomic focuses as detailed in Section 3.2.

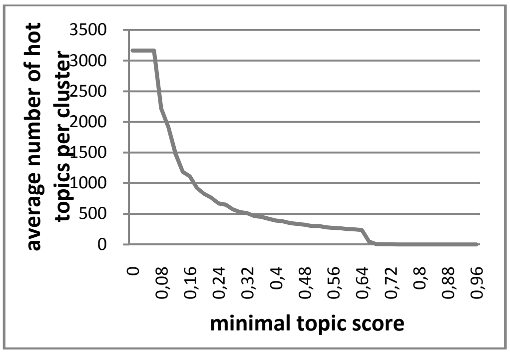

Figure 1: Average number of hot topics per cluster given different minimal topic score (ζ).

3.4 Hybrid Taxonomy Recommender

In this section, we describe the proposed Hybrid Taxonomy based Recommender (HTR) that incorporates the hot topic set described in Section 3.3 with the item-based collaborative filtering (item-based CF) to improve recommendation quality.

HTR generates item recommendations by combining the estimates to item preferences and the estimates to taxonomy preferences. We firstly explain the item-based CF technique used in HTR to estimate item preferences. Item-based CF recommends item t to user u based on the item similarity between t and the items that have been rated by u based on user ratings to these items. The similarity between two items is computed based on user explicit ratings as defined below:

$$item\_sim(t_i, t_j) = \frac{\sum_{u \in U_{ij}} (r_u^i - \bar{r_i})(r_u^j - \bar{r_j})}{\sqrt{\sum_{u \in U_{ij}} (r_u^i - \bar{r_i})^2}\sqrt{\sum_{u \in U_{ij}} (r_u^j - \bar{r_j})^2}} \quad (6)$$

where r_u^i is a simplified form for $rating(u, t_i)$ representing user u's rating to item t_i, $\bar{r_i}$ is the average rating for t_i' over the users in U_{ij}, and U_{ij} is the set of users who have rated both t_i and t_j. U_{ij} is defined as:

$$U_{ij} = \{u \in U | \{t_i, t_j\} \subset RT_{expl}(u)\}$$

Note, it is possible that two items are never rated by more than one user, i.e. $U_{ij} = \emptyset$. In such case, $item\_sim(t_i, t_j)$ returns a special value NC which is a label indicating “Not Computable”.

As mentioned above, the estimate of the preference to item t to user u is based on the similarities between t and the items $t' \in RT_{expl}(u)$ rated by the user u, where $t \neq t'$. In order to achieve it, we need to find the target user’s rated items which are computable with the target item t. That is,

$$cItems(u, t) = \{t' \in RT_{expl}(u) | item\_sim(t, t') \neq NC\}$$

Finally, user u’s item preference prediction to item t is computed as below:

$$\eta_{u,t} = \frac{\sum_{y \in cItems(u,t)} (item\_sim(y, t) \times r_u^y)}{\sum_{y \in cItems(u,t)} |item\_sim(y, t)|} \quad (7)$$

where $0 \leq \eta_{u,t} \leq 1$.

In order to improve the recommendation quality (especially in cold start situations), HTR also checks whether the taxonomy of the candidate items is preferred by the target user. We use $\psi_{u,t}$ to denote the prediction of user u’s taxonomic preference to item t, and it can be computed as below:

$$\Psi_{u,t} = \begin{cases} max_{p \in \delta}\, topic_{score(p, cluster(u))}, & |\delta| > 0 \\ 0, & otherwise \end{cases} \quad (8)$$

where

$\delta = \langle \{p | p \in d, d \in D(t)\} \cap hot\_topics(cluster(u), \zeta) \rangle$

is the set of t’s topics that are hot topics of the cluster which contains u. The idea behind the computation of taxonomic preference score is straightforward. We firstly check if any of the target item t’s taxonomy topics are hot topics of the user u’s neighbourhood (i.e. $cluster(u)$). If the item’s topics are not hot topic of $cluster(u)$, then we suggest that the user is not interested in the item’s taxonomy, hence 0 will be given as the taxonomy score. If the item’s topics are in the hot topic set, then among these matched hot topics ($|\delta|$ can be greater than 1), the maximum hot topic score is chosen as t’s taxonomy score.

It should be mentioned that the hot topics calculated by Equation (5) represent the taxonomic focuses of the users in a cluster. That means the topics in δ represent cluster level taxonomic focuses commonly preferred by the users in that cluster but not particularly for any individual user. There are two reasons for doing so. Firstly, cluster level taxonomic preferences can be pre-computed offline, therefore it ensures the computation efficiency of the proposed technique. Secondly, since the cluster level taxonomic preferences cover the taxonomic interests of all the users in one cluster, for the target user, by recommending items with topics commonly preferred by the users in the cluster, the recommender can recommend items with a wider range of topics including the topics which may not be particularly preferred by the target user but preferred by the users in this cluster and thus the recommendation quality can be improved.

In order to recommend a set of k items to a target user $u \in U$, we firstly form a candidate item list containing all items rated by u's neighbors but not yet rated by u. Next, for each item t in the candidate list, we compute the item preference score and taxonomic preference score for the item. The proposed preference score for each candidate item can then be computed by combining the item preference score ($\eta_{u,t}$) and the item taxonomic preference score ($\Psi_{u,t}$) together. Finally, k candidate items with highest preference scores are recommended to the user u, and these recommended items are sorted by the ranking values. The complete algorithm is listed below:

Algorithm $taxonomy\_recommender(u, k)$.
where $u \in U$ is a given target user
k is the number of items to be recommended

1) SET $\gamma_u = [\cup_{w \in cluster(u)} RT_{all}(w)] \backslash RT_{all}(u)$, the candidate item list
2) FOR EACH $t \in \gamma_u$
3) SET $rank_{u,t} = \alpha\eta_{u,t} + (1-\alpha)\Psi_{u,t}$
4) END FOR
5) Return the top k items with highest $rank_{u,t}$ scores to u.

From line (3) of the algorithm we can see that the predicted score for an item is computed by a linear combination of item preference score $\eta_{u,t}$ and topic preference score $\Psi_{u,t}$. The coefficient α, computed by Equation (9) below, in the formula is used to adjust the weights of $\eta_{u,t}$ and $\Psi_{u,t}$:

$$\alpha = \frac{\omega\vartheta}{\omega\vartheta + (1-\omega)(1-\vartheta)} \quad (9)$$

where $\omega = \frac{|cItems(u,t)|}{|RT_{expl}(u)|}$ and $0 \le \vartheta \le 1$ is a user controlled variable. ω is the ratio between the number of the items that are commonly rated with item t by u and other users and the number of the items rated by u.

In Equation (9), ω reflects the quality confidence of $\eta_{u,t}$, because the more the target user's past rated items related to the target item, the higher the accuracy of the item preference prediction (i.e. $\eta_{u,t}$) will be. When ω increases α will increase too, thus $\eta_{u,t}$ will receive higher weight in the final score (i.e. $rank_{u,t}$). Variable ϑ, on the other hand, is used to adjust the weights of ω in α, thus, if ϑ is large (e.g. 0.9) $\eta_{u,t}$ will still receive high weight even ω is small.

The value of α is automatically adjusted along with the change of the number of users who commonly rated a given item t. The higher the value of α the more the users who commonly rated the item (i.e., ω is high which indicates a normal situation without severe cold start problems) and thus the item preference $\eta_{u,t}$ estimated based on these users' rating data becomes more important and reliable. In this case, the predicted item preference $\eta_{u,t}$ makes more contributions to the predicted score $rank_{u,t}$ to item t than the contribution made by the predicted taxonomic preference $\Psi_{u,t}$. On the other hand, if the value of α is low (i.e. ω is low which indicates a cold start situation), the taxonomic preference prediction becomes more important and will contribute more to the predicted score $rank_{u,t}$ that what the predicted item preference does. This design ensures that taxonomic preferences are used to supplement or enrich the item preference prediction, especially in cold start situations.

4 EXPERIMENTATION

This section presents empirical results obtained from our experiment.

4.1 Data Acquisition

The dataset used in this experiment is the "Book-Crossing" dataset (http://www.informatik.uni-freiburg.de/~cziegler/BX/), which contains 278,858 users providing 1,149,780 ratings about 271,379 books. In the user ratings, 433,671 of them are the explicit user ratings, and the rest of 716,109 ratings are implicit ratings.

The taxonomy tree and book descriptors for our experiment are obtained from Amazon.com. Amazon.com's book classification taxonomy is tree-structured (i.e. limited to "single inheritance") and therefore is perfectly suitable to the proposed technique. However, not every book in our dataset is available in Amazon.com, and we were only able to extract taxonomy descriptors for 270,868 books form Amazon.com. The books without descriptors are removed from the dataset. The average number of descriptors per book is around 3.15, and the taxonomy tree formed by these descriptors contains 10746 unique topics.

4.2 Experiment Framework

All recommenders being used in the experiment are developed using the Taste (http:// taste.sourceforge.net/) framework. Taste provides a set of standardized components for developing recommenders, therefore it ensures the comparability of the developed recommenders fairly. Moreover, Taste also provides an evaluation framework allowing researchers or developers to evaluate the performances of their recommenders with a standardized test bed easily and effectively.

In this experiment we constructed 7 different recommenders, and they are listed in Table 2.

Table 2: List of experimental recommenders.

Type / Name	Descriptions
Item based Recommender **[IR]**	Standard item-based CF, the detailed algorithm is listed in (Deshpande and Karypis 2004).
Item based Recommender with User Clustering **[IRC]**	Standard item-based CF, however this version only recommend items within the candidate item list γ_u in order to improve computation efficiency.
Slop One Recommender **[SO]**	A well known modern item based recommendation technique(Lemire and Maclachlan 2005), it features on its implementation simplicity and computation efficiency.
Taxonomy Product Recommender **[TPR]**	A taxonomy based recommender proposed by Ziegler(Ziegler et al. 2004). This work uses similar taxonomy scheme to our work, and therefore can be a good benchmark.
Item based Recommender with TPR **[ITR]**	The combination of the item-based CF and TPR. The hybridization scheme is identical to HTR. The only difference is that $\Psi_{u,t}$ is computed using Ziegler's method.
Hybrid Taxonomy Recommender **[HTR]**	The proposed HTR method using users' explicit rating data and implicit rating data as well
Hybrid Taxonomy Recommender (with only explicit ratings) **[HTR_E]**	The proposed HTR method using only explicit ratings. The purpose of conducting this test is to ensure fair comparison with IR, IRC, SO, which use only explicit ratings.

4.3 Evaluation Metrics

The goal of our experiment in this paper is to compare the recommendation performances and computation efficiencies for the recommenders listed in Table 2.

For the recommendation quality evaluation, we randomly divided each user $u_i \in U$'s past ratings (i.e. $RT_{all}(u_i)$) into two parts, one for training and another for testing. We use R_i to denote u_i 's training rating data and T_i to denote the testing rating data, such that $R_i \cup T_i = RT_{all}(u_i)$, $R_i \cap T_i = \emptyset$, and $|R_i| \cong |T_i|$. The testing data T_i actually consists of three types of items, and they are:

- Items implicitly rated by u_i: $\dot{T}_i = T_i \cap RT_{impl}(u_i)$
- Items preferred by u_i: $\check{T}_i =$
  $\{t | t \in T_i \cap RT_{expl}(u_i), rating(u_i, t) > avg(u_i)\}$
- Items not preferred by u_i: $\tilde{T}_i = \{T_i \cap RT_{expl}(u_i)\} \setminus \check{T}_i$

In the experiment, the recommenders recommend a list of k items P_i to u_i based on the training set R_i, and the recommendation list P_i can be evaluated with $\check{T}_i$. In order to evaluate the performances of different recommenders based on $\check{T}_i$ and P_i, recommendation list based evaluation metrics such as precision and recall, Breese Score, Half-life, and etc. (Herlocker et al. 2004; Schein et al. 2002) can be utilized. In this paper, the precision and recall metric is used for the evaluation, and its formulas are listed below:

$$Recall = \frac{|\check{T}_i \cap P_i|}{|\check{T}_i|} \quad (10)$$

$$Precision = \frac{|\check{T}_i \cap P_i|}{|P_i|} \quad (11)$$

In order to provide a general overview of the overall performances, F1 metric is used to combine the results of Precision and Recall:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (12)$$

For the computation efficiency evaluation, the average time required by recommenders to make a recommendation will be compared.

4.4 Experiment Result

The test dataset is constructed by randomly choosing 10,000 users from the 278,858 users in the Book-Crossing dataset mentioned in Section 4.1. We let each recommenders recommend a list of k items to these 10,000 users. We tested different values for k ranging from 5 to 25.

The results of this part of the experiment are shown in Figure 2, Figure 3 and Figure 4. It can be observed from the figures that, for all the three evaluation metrics the proposed HTR technique achieves the best result among all the recommenders. In the case of using only explicit rating data, the recommendation quality of HTR (i.e. HTR_E) still outperforms other recommenders even slightly

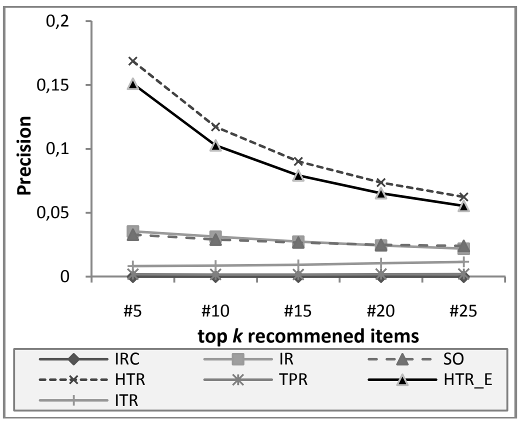

Figure 2: Recommender evaluation with precision metric.

degrading compared with using both explicit and implicit rating data (i.e., HTR performs the best and HTR_E performs the second best.

The standard item based CF recommender (IR) performed similarly to the slope one recommender (SO), however it seems that slope one recommender is slightly better in recommending longer item lists.

In the experiment, the clustering-based CF recommender (IRC) performed better than the standard one (IR). The only difference between these two recommenders is in the candidate item list formation process. The standard item based CF uses all items from the dataset as its candidate item list (i.e. $T\backslash RT_{all}(u)$), whereas the clustering-based version uses only items within a user cluster (i.e. $\left[\bigcup_{w\in cluster(u)} RT_{all}(w)\right]\backslash RT_{all}(u)$). Intuitionally, the clustering-based CF might perform worse than the standard one, because its candidate item list is formed from a cluster which is only a subset of the entire item set, some potential promising items might be excluded and thus won't be recommended. However, based on our observation, many of these excluded items are noises generated from the item similarity measure (some item similarity measures might generate prediction noise, please refer to (Deshpande and Karypis 2004) for more information), therefore by removing these items from the candidate list can actually improve the recommendation quality. The proposed HTR also gets benefits from the clustering strategy as it generates recommendations from the candidate item list formed from a cluster.

We also implemented the TPR technique proposed by Ziegler(Ziegler et al. 2004), and it performed worst among all recommenders in our evaluation scheme. TPR uses only implicit ratings as its data source and generates recommendations only based on taxonomy preferences. In order to make the proposed HTR and Ziegler's TPR more comparable, we modified TPR by adding the item-based CF component into TPR resulting in the new recommender ITR. ITR performed better than the standard TPR as it included the item preference consideration in its recommendation making process. However it is still worse than all other recommenders (i.e., TPR performs the worst and ITR performs the second worst). The difference between HTR and ITR is the method to compute the taxonomy preferences is different (they use the same method to compute the item preferences). The result of HTR outperforming ITR indicates that users' item preference is also helpful for generating users' taxonomy preference. The proposed HTR technique considers the item preference implication when generating the taxonomic preferences (i.e. the taxonomic preferences are extracted from user clusters which is divided based on users' item preferences). In contrary, TPR generates users' taxonomic preferences purely from taxonomy data without using any of the users' item preferences.

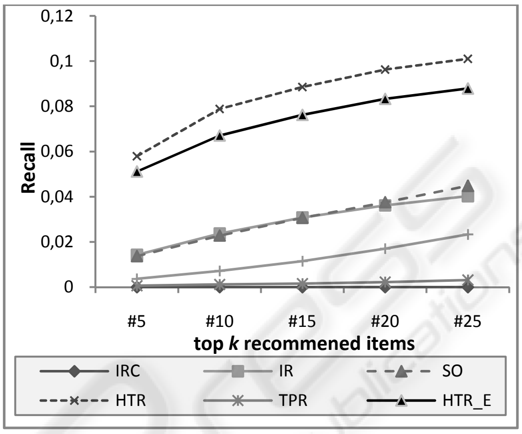

Figure 3: Recommender evaluation with recall metric.

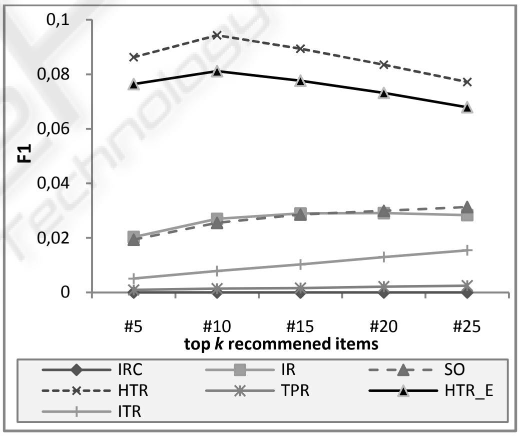

Figure 4: Recommender evaluation with F1 metric.

In the experiment, the recommender with the best computation efficiency is the clustering based CF (IRC) as showed in Figure 5, it is much faster than the standard CF because its candidate item list is much smaller. The proposed HTR methods (HTR and HTR_E) perform the third and second best, as they added a bit computation complexity in the taxonomic preference predictions. However, this extra computation complexity is trivial, because most of these computations (i.e. computing hot_topics for each user cluster) can be done offline. HTR_E performed slightly better than HTR because it uses less data (only explicit ratings) to make recommendations. Ziegler's TPR is computation expensive because it needs to convert all users and items into high dimensional taxonomy vectors. ITR performed slightly worse than TPR because it needs to compute extra item preference predictions using standard CF technique. Standard CF technique is the most inefficient one among all the recommenders, whereas slop one recommender offers a slight advantage in computation efficiency.

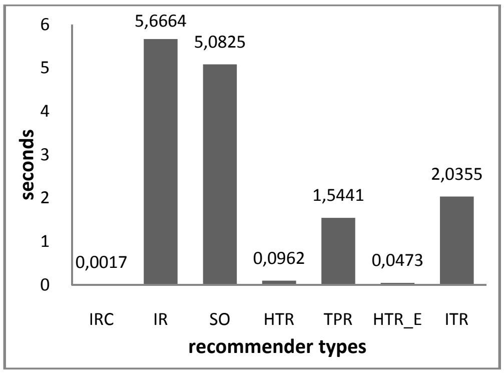

Figure 5: Average second per recommendation.

5 CONCLUSIONS

In this paper, we investigated the implicit relations between users' item preferences and taxonomic preferences, suggested and also verified using information gain that users that share similar item preferences may also share similar taxonomic preferences. Based on this investigation, we proposed a novel, hybrid technique HTR to automated recommendation making based upon large-scale item taxonomies which are readily available for diverse ecommerce domains today.

HTR produces quality recommendations by incorporating both users' taxonomic preferences and item preferences. Moreover, it can utilize both explicit and implicit ratings for recommendation making, and hence they are less prone to the cold start problem. We have compared the proposed HTR technique with some standard benchmark techniques such as item-based recommender and some advanced modern techniques such as TPR (which are related to ours). We have conducted extensive experiments which demonstrated that the proposed HTR outperforms other recommenders in both recommendation quality and computation efficiency.

REFERENCES

Adomavicius, Gediminas, Ramesh Sankaranarayanan, Shahana Sen and Alexander Tuzhilin. 2005. "Incorporating contextual information in recommender systems using a multidimensional approach " ACM Trans. Inf. Syst. 23(1):103-145.

Breese, J. S., D. Heckerman and C. Kadie. 1998. "Empirical Analysis of Predictive Algorithms for Collaborative Filtering." In Proceedings of 14th Conference on Uncertainty in Artificial Intelligence. Madison, WI.

Burke, Robin. 2002. "Hybrid Recommender Systems: Survey and Experiments." User Modeling and User-Adapted Interaction 12(4):331-370.

Deshpande, Mukund and George Karypis. 2004. "Item-based top-N recommendation algorithms." ACM Transactions on Information Systems 22(1):143-177.

Ferman, A. Mufit, James H. Errico, Peter van Beek and M. Ibrahim Sezan. 2002. "Content-based filtering and personalization using structured metadata." In 2nd ACM/IEEE-CS joint conference on Digital libraries Portland, Oregon, USA.

Herlocker, Jonathan L., Joseph A. Konstan, Loren G. Terveen and John T. Riedl. 2004. "Evaluating collaborative filtering recommender systems." ACM Transactions on Information Systems (TOIS) 22(1):5-53.

Lemire, D. and A. Maclachlan. 2005. "Slope One Predictors for Online Rating-Based Collaborative Filtering." In 2005 SIAM Data Mining

Middleton, Stuart E, Harith Alani, Nigel R Shadbolt and David C. De Roure. 2002. "Exploiting Synergy Between Ontologies and Recommender Systems." In The Semantic Web Workshop, World Wide Web Conference.

Montaner, Miquel, Beatriz López and Josep Lluís De La Rosa. 2003. "A Taxonomy of Recommender Agents on the Internet." Artificial Intelligence Review 19(4):285-330.

Park, Seung-Taek, David Pennock, Omid Madani, Nathan Good and Dennis DeCoste. 2006. "Naive filterbots for robust cold-start recommendations." In 12th ACM SIGKDD international conference on Knowledge discovery and data mining, ed. ACM Press. Philadelphia, PA, USA.

Sarwar, B., G. Karypis, J. Konstan and J. Riedl. 2000. "Application of dimensionality reduction in recommender systems--a case study." In ACM WebKDD Workshop. Boston, MA, USA.

Schafer, J. Ben, Joseph A. Konstan and John Riedl. 2000. "E-Commerce Recommendation Applications." Journal of Data Mining and Knowledge Discovery 5:115-152.

Schein, Andrew I., Alexandrin Popescul, Lyle H. Ungar and David M. Pennock. 2002. "Methods and metrics for cold-start recommendations" In 25th annual international ACM SIGIR conference on Research and development in information retrieval. Tampere, Finland: ACM Press.

Ziegler, Cai-Nicolas, Georg Lausen and Lars Schmidt-Thieme. 2004. "Taxonomy-driven Computation of Product Recommendations" In International Conference on Information and Knowledge Management Washington D.C., USA