PERGAMON

Expert Systems with Applications 26 (2004) 233–246

Expert Systems with Applications

www.elsevier.com/locate/eswa

Application of Web usage mining and product taxonomy to collaborative recommendations in e-commerce

Yoon Ho Cho[a,1], Jae Kyeong Kim[b,*]

[a]Department of Internet Information, Dongyang Technical College, 62-160 Kochuk-dong, Kuro-gu, Seoul 152-714, South Korea
[b]School of Business Administration, Kyung Hee University, #1, Hoeki-dong, Dongdaemoon-gu, Seoul 130-701, South Korea

Abstract

The rapid growth of e-commerce has caused product overload where customers on the Web are no longer able to effectively choose the products they are exposed to. To overcome the product overload of online shoppers, a variety of recommendation methods have been developed. Collaborative filtering (CF) is the most successful recommendation method, but its widespread use has exposed some well-known limitations, such as sparsity and scalability, which can lead to poor recommendations. This paper proposes a recommendation methodology based on Web usage mining, and product taxonomy to enhance the recommendation quality and the system performance of current CF-based recommender systems. Web usage mining populates the rating database by tracking customers' shopping behaviors on the Web, thereby leading to better quality recommendations. The product taxonomy is used to improve the performance of searching for nearest neighbors through dimensionality reduction of the rating database. Several experiments on real e-commerce data show that the proposed methodology provides higher quality recommendations and better performance than other CF methodologies.

Keywords: Collaborative filtering; Internet marketing; Personalized recommendation; Product taxonomy; Web usage mining

1. Introduction

Nowadays, an unprecedented number of companies are using the Internet to market and sell products (Sarwar, 2001). This movement toward e-commerce has allowed companies to provide customers with more choices on products. However, increasing choice has also caused product overload where the customer is no longer able to effectively choose the products he/she is exposed to. A promising technology to overcome the product overload problem is recommender systems that help customers find the products they would like to purchase. To date, a variety of recommendation techniques have been developed. Collaborative filtering (CF) is the most successful recommendation technique, which has been used in a number of different applications such as recommending movies, articles, products, Web pages, etc. (Balabanović & Shoham, 1997; Basu, Hirsh, & Cohen, 1998; Billsus & Pazzani, 1998; Cho, Kim, & Kim, 2002; Claypool et al., 1999; Hill, Stead, Rosenstein, & Furnas, 1995; Kim, Cho, Kim, Kim, & Suh, 2002; Lawrence, Almasi, Kotlyar, Viveros, & Duri, 2001; Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994; Shardanand & Maes, 1995; Soboroff & Nicholas, 1999; Terveen, Hill, Amento, McDonald, & Creter, 1997).

CF-based recommender systems recommend products to a target customer according to the following steps (Sarwar, Karypis, Konstan, & Riedl, 2000b, 2001): (1) A customer provides the system with preference ratings of products that may be used to build a *customer profile* of his or her likes and dislikes. (2) Then, these systems apply statistical techniques or machine learning techniques to find a set of customers, known as *neighbors*, which in the past have exhibited similar behavior (i.e. they either rated similarly or purchased similar set of products). Usually, a neighborhood is formed by the degree of similarity between the customers. (3) Once a neighborhood of similar customers is formed, these systems predict whether the target customer will like a particular product by calculating a weighted composite of the neighbors' ratings of that product (*prediction problem*), or generate a set of products that the target customer is most

* Corresponding author. Tel.: +82-2-961-9355; fax: +82-2-967-0788.
E-mail addresses: jaek@khu.ac.kr (J.K. Kim); yhcho@dongyang.ac.kr (Y.H. Cho).
[1] Tel.: +82-2-2610-1918; fax: +82-2-2610-1859.

0957-4174/$ - see front matter © 2003 Elsevier Ltd. All rights reserved.
doi:10.1016/S0957-4174(03)00138-6

likely to purchase by analyzing the products the neighbors purchased (*top-N recommendation problem*). These systems, also known as the *nearest neighbor* CF-based recommender systems (Breese, Heckerman, & Kadie, 1998; Herlocker, Konstan, Borchers, & Riedl, 1999; Sarwar et al., 2000b) have been widely used in practice. However, as the number of customers and that of products managed in an e-commerce site grow rapidly, its application to e-commerce has exposed two major issues that must be addressed (Claypool et al., 1999; Sarwar, 2001; Sarwar, Karypis, Konstan, & Riedl, 2000a; Sarwar et al., 2000b, 2001).

The first issue is related to *sparsity*. In a large e-commerce site such as Amazon.com, there are millions of products and so customers may rate only a very small portion of those products. Most similarity measures used in CF work properly only when there exists an acceptable level of ratings across customers in common. Such sparsity in ratings makes the formation of neighborhood inaccurate, thereby resulting in poor recommendation. Many approaches have been proposed to overcome the sparsity problem. These approaches can be classified into three categories: implicit ratings, hybrid filtering and product-to-product correlation. The implicit ratings approaches attempt to increase the number of ratings through observing customers' behavior (Good et al., 1999; Konstan et al., 1997; Rucker & Polano, 1997; Sarwar et al., 1998; Terveen et al., 1997). The hybrid filtering approaches combine content-based filtering and CF for augmenting sparse preference ratings (Aggarwal & Yu, 2000; Balabanović & Shoham, 1997; Basu et al., 1998; Claypool et al., 1999; Condliff, Lewis, Madigan, & Posse, 1999; Melville, Mooney, & Nagarajan, 2001; Mobasher, Dai, Luo, Sun, & Zhu, 2000; Soboroff & Nicholas, 1999). These approaches learn to predict which products a given customer will like by matching properties associated with each product to those associated with products that he/she has liked in the past, and then use such a content-based prediction to convert a sparse customer profile into a dense one. Instead of identifying the neighborhood of similar customers, the product-to-product correlation approach analyzes the customer profile to identify relationships between different products and then uses these relations to compute the prediction score for a given customer–product pair (Sarwar et al., 2001).

The second issue is related to *scalability*. Recommender systems for large e-commerce sites have to deal with millions of customers and products. Because these systems usually handle very high-dimensional profiles to form the neighborhood, the nearest neighbor algorithm is often very time-consuming and scales poorly in practice (Sarwar et al., 2000b). To address the scalability problems in CF-based recommender systems, a variety of approaches have been developed. These approaches can be classified into two main categories: dimensionality reduction techniques and model-based approaches (Sarwar, 2001). Latent Semantic Index (LSI) is a widely used dimensionality reduction technique. It uses singular value decomposition (SVD) to factor the original rating space into three matrices and performs the dimensionality reduction by reducing the singular matrix (Billsus & Pazzani, 1998; Sarwar et al., 2000a). In model-based approaches, a model is first built based on the rating matrix and then the model is used in making recommendations. Usually, the model is expensive to build, but rapid to execute. Several data mining techniques such as Bayesian network (Breese et al., 1998; Hoffman & Puzicha, 1999), clustering (Basu et al., 1998; Breese et al., 1998; Ungar & Foster, 1998) and association rule mining (Lin, Alvarez, & Ruiz, 2000, 2002; Sarwar et al., 2000b) have been applied to building the model.

In this paper, we propose a recommendation methodology, called Web usage mining driven CF recommendation methodology using Product Taxonomy (WebCF-PT), to address the sparsity and scalability problems of current CF-based recommender systems.

Web usage mining is employed as an implicit ratings approach to address the sparsity problem. Web usage mining analyzes customers' shopping behaviors on the Web and collects their implicit ratings. This increases the number of ratings rather than only collecting explicit ratings, thereby reducing the sparsity. E-commerce data are rich and detailed compared to off-line commerce data. One type of collected e-commerce data is a clickstream that tracks visitors' path through a Web site (Lee, Podlaseck, Schonberg, & Hoch, 2001). The clickstream in Web retailers provides information essential to understand the shopping patterns or prepurchase behaviors of customers such as what products they see, what products they add to the shopping cart, and what products they buy. By analyzing such information via Web usage mining, it is possible not only to make a more accurate analysis of the customer's interest or preference across all products (than analyzing the purchase records only), but also to increase the number of ratings (when compared to collecting explicit ratings only). Nevertheless, the existing research in recommender systems has not offered a formal way for capturing implicit ratings of individual customer through Web usage mining. In this paper, we suggest a formal scheme to capture implicit ratings by analyzing customers' online shopping behaviors and to build the customer profiles.

To solve the scalability problem, we use a nearest neighbor CF algorithm. But, before applying the algorithm, we reduce the dimensionality of the customer profiles. As a dimensionality reduction technique, we employ a product taxonomy that represents hierarchical relationships between products as domain specific knowledge provided by marketing managers or domain experts. Similar products are identified and they are grouped together using the product taxonomy so as to build the customer profiles and to search for the neighbors in the reduced dimensional space. This product grouping thus leads to reducing the sparsity of preference ratings as well as improving the scalability of making recommendations.

We begin by reviewing the concept of Web usage mining and product taxonomy which is prerequisite for our research in Section 2. In Section 3, WebCF-PT is explained step by step, and experimental results and discussions are provided in Section 4. Finally, Section 5 concludes this paper with suggestions for future research.

2. Related work

2.1. Web usage mining

Web usage mining is the process of applying data mining techniques to the discovery of behavior patterns based on Web log data, for various applications. In the advance of e-commerce, the importance of Web usage mining grows larger than before. The overall process of Web usage mining is generally divided into two main tasks: data preprocessing and pattern discovery. Mining behavior patterns from Web log data needs the data preprocessing tasks that include data cleansing, user identification, session identification, and path completion. Data cleansing performs merging Web logs from multiple servers, removing irrelevant and redundant log entries with filename suffixes such as gif, jpeg, map, count.cgi, etc. and parsing of the logs. To track individual user's behaviors at a Web site, user identification and session identification is required. For Web sites using session tracking such as URL rewriting, persistent cookies or embedded session IDs, user and session identification is trivial. Web sites without session tracking must rely on heuristics. Path completion may also be necessary because of local or proxy level caching. Cooley, Mobasher, and Srivastava (1999) presented a detailed description of data preprocessing methods for mining Web browsing patterns. The pattern discovery tasks involve the discovery of association rules, sequential patterns, usage clusters, page clusters, user classifications or any other pattern discovery method (Mobasher, Cooley, & Srivastava, 2000). The usage patterns extracted from Web data can be applied to a wide range of applications such as Web personalization, system improvement, site modification, business intelligence discovery, usage characterization, etc. (Srivastava, Cooley, Deshpande, & Tan, 2000).

Our methodology recommends products based on Web usage data as well as product purchase data and customer-related data. There have been several customer behavior models for e-commerce, which have different analysis purposes. Menascé, Almeida, Fonseca, and Mendes (1999) presented a state transition graph, called Customer Behavior Model Graph (CBMG) to describe the behavior of groups of customers who exhibit similar navigational patterns. VandeMeer, Dutta, and Datta (2000) developed a user navigation model designed for supporting and tracking dynamic user behavior in online personalization. Lee et al. (2001) provided a detailed case study of clickstream analysis from an online retail store. To measure the effectiveness of efforts in merchandising, they analyzed the shopping behavior of customers according to the following four shopping steps: product impression, click-through, basket placement, and purchase. A part of Lee et al.'s model was adopted for our research, because they focus the case of Web retailer which is also our consideration.

2.2. Product taxonomy

In most Web retailers, product taxonomy is available. A product taxonomy is practically represented as a tree that classifies a set of products at a low level into a more general product at a higher level. The leaves of the tree denote the *product instances*, Stock Keeping Units (SKUs) in retail jargon, and non-leaf nodes denote *product classes* obtained by combining several nodes at a lower level into one parent node. The root node labeled by *All* denotes the most general product class. Fig. 1 shows an example product taxonomy for a fashion Web retailer, where *Outerwear, Pants* and *Shirts* are classified into *Clothes*, etc.

A number called *level* can be assigned to each node in the product taxonomy. The level of the root node is zero, and the level of any other node is one plus the level of its parent. Note that a product class at a higher level has a smaller level

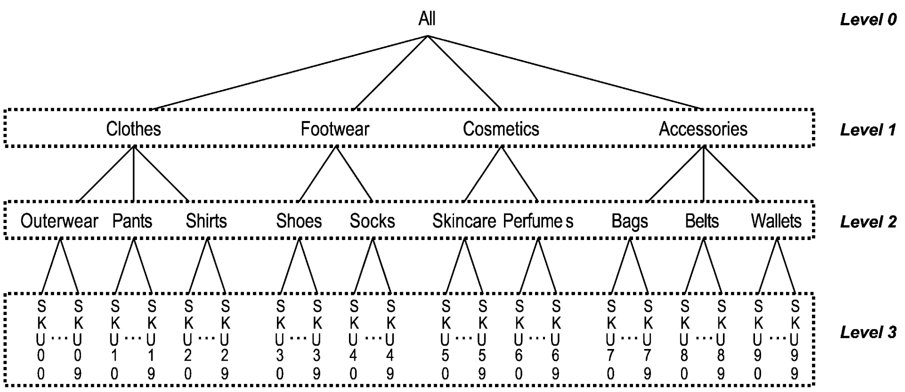


Fig. 1. Example of a product taxonomy.

number. The product taxonomy of Fig. 1 has four levels, referred to as level 0 (for root), 1, 2, and 3.

Recently, the usage of product taxonomy in data mining has been emphasized by many researchers (Adomavicius & Tuzhilin, 2001; Berry & Linoff, 1997; Han & Fu, 1995, 1999; Lawrence et al., 2001) since it reflects domain specific knowledge and may affect the results of the analysis. Adomavicius and Tuzhilin (2001) proposed a useful way for the domain expert to examine multiple rules at a time by grouping similar rules together on given product taxonomy. Berry and Linoff (1997) stressed that it is important to mine association rules at the right level of the product taxonomy. Han and Fu (1995, 1999) also discussed that mining association rules at different levels of the product taxonomy may lead to the discovery of more specific and concrete knowledge rather than at a single level. Lawrence et al. (2001) used the product taxonomy to capture the affinity between different products in developing a supermarket product recommender system.

3. Methodology

3.1. Overall procedure

WebCF-PT is a recommendation methodology based on Web usage mining and the product taxonomy to improve the recommendation quality and the system performance of current CF-based recommender systems. The overall procedure of WebCF-PT is divided into four phases as shown in Fig. 2: grain specification, customer profile creation, neighborhood formation, and recommendation generation.

The input data consist of Web server log files, product database, customer database and purchase database. The endmost output is the personalized product recommendation list. Using the recommendation list, the Web retailer may be able to perform effective one-to-one marketing campaigns of providing individual target customer with a personalized product recommendation by the delivery of e-mail or by the presence of personalized Web pages, etc. In the grain specification phase, all products in the database are hierarchically grouped based on the level of aggregation (called *grain*) specified from the marketing managers. Such a product grouping enables the following phases to handle products in the reduced dimensional space. Target customer's preferences across products are analyzed and used to make customer profiles in the customer profile creation phase. Tracking individual customer's previous shopping behavior in an e-commerce site is used to make preference analysis. The neighborhood formation phase is to form a similarity-based neighborhood between a target customer and a number of like-minded customers. Finally, the recommendation generation phase produces the *top-N* recommendations based on the shopping behavior of neighbors. A more detailed description for each phase is provided in the following subsections.

3.2. Phase 1: grain specification

In this phase, similar products are identified and they are grouped together using product taxonomy so as to conduct the following phases in the reduced product space. The marketing manager or domain expert categorizes all the products by specifying the level of product aggregation on the product taxonomy. We refer to such a specification as *grain*. We introduce the concept of grain from the concept of 'cut' which Adomavicius and Tuzhilin (2001) proposed as a flexible way for the domain expert to examine multiple

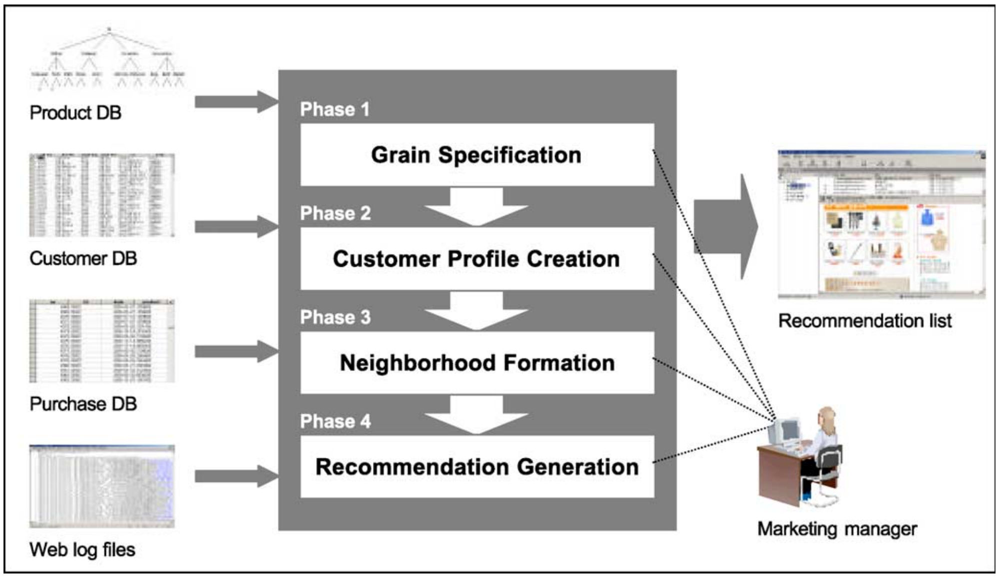


Fig. 2. Overall procedure of WebCF-PT.

rules at a time by grouping similar rules together. Informally, a grain G is a subset of all the nodes (product and product class) of the product taxonomy except the root node, such that for every path from a leaf node (product) to the root node, one and only one node on such path belongs to this subset. Therefore, every leaf node has its corresponding grain node (it is called as a *grain product class* in this paper). Given a grain G, we define for any product x its corresponding grain product class $\text{class}_G(x)$ as follows:

$$\text{class}_G(x) = \begin{cases} x, & \text{if } x \in G, \\ \text{class}_G(\text{parent}(x)), & \text{otherwise.} \end{cases} \quad (1)$$

Consider several examples of specifying the grain as shown in Fig. 3, where grains are denoted by shaded regions. Fig. 3(a) shows a case of specifying the grain at a lower level, such that $G = \{$Outerwear, Pants, Shirts, Shoes, Socks, Skincare, Perfumes, Bags, Belts, Wallets$\}$. Fig. 3(b) shows a case of specifying the grain at a higher level, such that $G = \{$Clothes, Footwear, Cosmetics, Accessories$\}$. For the grain from Fig. 3(a), $\text{class}_G(\text{SKU00}) = \text{Outerwear}$ and $\text{class}_G(\text{SKU10}) = \text{Pants}$. However, in the grain of Fig. 3(b), $\text{class}_G(\text{SKU00}) = \text{Clothes}$ and $\text{class}_G(\text{SKU10}) = \text{Clothes}$. The grain can be specified across different levels of the taxonomy as shown in Fig. 3(c). In this specification, the proper grain level depends on the product, on its relative importance for making recommendations and on its frequency in the transaction data. For example, frequently purchased products might stay at a low level in the taxonomy while rarely purchased products might be drilled up to higher levels.

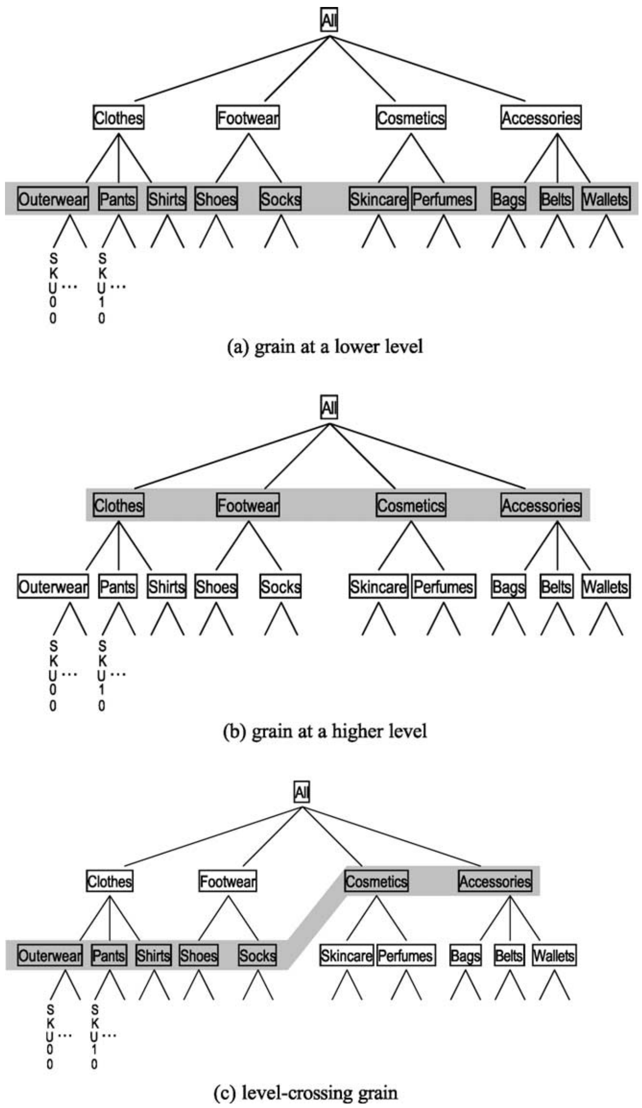


Fig. 3. Examples of specifying the grain.

Specifying the grain at different levels would give different recommendation results, as the higher leveled grain treats the transaction data in more aggregated way. Hence, the recommendation based on the right grain is expected to solve the sparsity and scalability problem in CF.

Although the proper grain can be specified by marketing managers or domain experts considering their previous experience or available transaction distribution statistics, a heuristic algorithm is necessary to find the right grain systematically. Recent data mining research has shown that data mining algorithms usually produce the best results when product-related transactions are evenly occurred (Berry & Linoff, 1997; Han & Fu, 1994). In our domain, the even transaction distribution among nodes in the grain is expected to prevent preference ratings from being dominated by the most frequent products. Hence, the basic idea of the heuristic algorithm is to reorganize or adjust the existing product taxonomy as a set of nodes with relatively even transaction distribution, i.e. not a blend of very big node and very small ones at the same level of the taxonomy, and then to specify the grain on the adjusted taxonomy. Han and Fu (1994) developed an efficient and effective algorithm that adjusts the taxonomy using the top-down big nodes promotion and bottom-up small nodes merging strategy. Based on their works, we define some terms and present an algorithm.

Let us define the following terms: the *prime level* is a level at which the grain is specified, the *prime node* is a node at the prime level, and the *dimensionality reduction threshold* is an integer restricting the number of distinct nodes in the grain (i.e. the number of product classes in the grain is no more than the given dimensionality reduction threshold).

Based on the above terms, we provide an algorithm for grain specification using the adjusted product taxonomy as shown in Fig. 4. The algorithm is designed based on the consideration that the nodes in the grain have to carry relatively even transaction distribution, simultaneously with the maximally preserved shape of the product taxonomy. The algorithm is expected to produce the desirable grain, and its performance is tested in subsequent section.

As an example, if a marketing manager wants to specify the grain with an even distribution among product classes using the original product taxonomy without adjustment, he/she may choose one of possible grains as shown in Fig. 5. This grain, specified across different levels, consists of seven product classes: {Outerwear(2340), Pants(531), Shirts(622), Shoes(1853), Socks(804), Cosmetics(2571), Accessories(1034)}, where each number in parentheses denotes the number of occurrences of sales in the corresponding product class. However, this is undesirable because each number of occurrences of sales in 'Pants',

```
procedure GrainSpecificationUsingAdjustedTaxonomy(T, D, P, L)
parameter
T: product taxonomy
D: dimensionality reduction threshold
P: purchase transactions set
L: prime level
begin

  // Initialization

  forall leaf node p in T do begin
    p.count ← the number of occurrences of sales for p in the sales transactions set P
  end
  Propagate the counts to the corresponding parents in the product taxonomy T
  total ← the sum of the counts of all the leaf nodes in the taxonomy T

  // Top-down promotion

  Grain ← Ø
  Buff ← {the root node}
  w ← 1 / D
  call CalculateWeight(0)
  call ExtendBuffer
  If |Grain| + |Buff| ≤ D then begin
    Move all the nodes in Buff to Grain
    Output Grain
    return
  end
  else begin
    do begin
      D' ← D - |Grain|
      w ← 1 / D'
      total ← the sum of counts of nodes in Buff
      call CalculateWeight(1)
      call ExtendBuffer
    until no change in Buff
  end

  // Bottom-up merging

  if Buff ≠ Ø then begin
    i ← L - 1
    while i ≥ 0 do begin
      for there are the nodes in Buff which share a common ancestor at level i then begin
        Merge the nodes into a new node m
        if m.weight ≥ w then begin
          Move m to Grain
        end
      end
      if |Buff| ≤ D' then begin
        Move all the nodes in Buff to Grain
        output Grain
        return
      end
```

Fig. 4. Algorithm for grain specification using the adjusted product taxonomy.

```
        else begin
          D' ← D - |Grain|
          w ← 1 / D'
          total ← the sum of counts of nodes in Buff
          call CalculateWeight(2)
          i ← i - 1
        end
      end
    end
    Output Grain
end

procedure CalculateWeight(F)
parameter
F: type of weight calculation
begin
    if F = 0 then begin
      forall non-leaf node p in T do begin
        p.weight ← p.count / total
        if p.weight ≥ w and p is a prime node then p.mark ← B
        else if p.weight ≥ w and p is not a prime node then p.mark ← B'
        else if p.weight < w and p is a prime node then p.mark ← S
        else if p.weight < w and p is not a prime node then p.mark ← S'
      end
    end
    else begin
      forall node p in Buff do begin
        p.weight ← p.count / total
      end
      if F = 2 then return
      forall node p in Buff do begin
        if p.weight ≥ w and p is a prime node then p.mark ← B
        else if p.weight ≥ w and p is not a prime node then p.mark ← B'
        else if p.weight < w and p is a prime node then p.mark ← S
        else if p.weight < w and p is not a prime node then p.mark ← S'
      end
    end
end

procedure ExtendBuffer
begin
    do begin
      forall node p in Buff do begin
        if p.mark = B then begin
          Move p in Buff to Grain
        end
        else if p.mark = B' then begin
          Replace p by its children
        end
      end
    until only the nodes marked S or S' are left in Buff
end
```

Fig. 4 (*continued*)

'Shirts' and 'Socks' is smaller than that of 'Outerwear' and 'Cosmetics'. It is desirable to merge the smaller nodes into an appropriately big node for discovering the grain with no blend of big node and small ones. When the prime level is 2 and the dimensionality reduction threshold is 6, the accomplishment of the algorithm results in Fig. 6, in which 'Pants' and 'Shirts' are merged together, and 'Socks' split from 'Footwear' is merged with 'Skincare' separated from 'Cosmetic'. The grain based on the adjusted product taxonomy consists of six product classes: {Outerwear(2340), Pants + Shirts(1153), Shoes(1853), Socks + Cosmetics(1579), Perfumes(1796), Bags + Belts + Wallets(1034)}, which show a relatively even distribution among all the product classes in the grain.

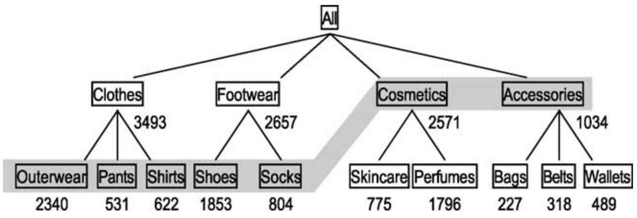


Fig. 5. Grain specification using the original product taxonomy.

3.3. Phase 2: customer profile creation

A customer profile is a description of customer interests or preferences about products. In CF, recommending products to a particular customer depends on his/her profile. This phase discovers customer's preferences across the grain specified at the previous phase, and makes the customer profile based on the preferences. For this purpose, the customer profile is constructed based on the following three general shopping steps in Web retailers, which is modified from the works of Lee et al. (2001):

(1) *click-through*: the click on the hyperlink and the view of the Web page of the product,
(2) *basket placement*: the placement of the product in the shopping basket,
(3) *purchase*: the purchase of the product—completion of a transaction.

A basic idea of measuring the customer's preference is simple and straightforward. The customer's preference is measured by counting the number of occurrence of URLs mapped to the product from clickstream of the customer. In Web retailers, products are purchased in accordance with the three sequential shopping steps: click-through, basket placement, and purchase. Hence, we can classify all products into four product groups such as purchased products, products placed in the basket, products clicked through, and the other products. This classification provides an *is-a* relation between different groups such that purchased products *is-a* products placed in the basket, and products placed in the basket *is-a* products clicked through. From this relation, it is reasonable to obtain a preference order between products such that {products never clicked} $\prec$ {products only clicked through} $\prec$ {products only placed in the basket} $\prec$ {purchased products}. Hence, it makes sense

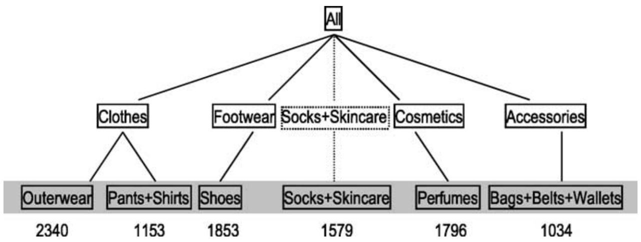


Fig. 6. Grain specification using the adjusted product taxonomy.

to assign the higher weight to occurrences of purchased products than those of products only placed in the basket. Similarly, the higher weight is given to products only placed in the basket than those of products only clicked through, and so on.

Let p_{ij}^c be the total number of occurrences of click-throughs of a customer i across every products in a grain product class j. Likewise, p_{ij}^b and p_{ij}^p are defined as the total number of occurrences of basket placements and purchases of a customer i for a grain product class j, respectively. p_{ij}^c, p_{ij}^b and p_{ij}^p are calculated from the raw clickstream data as the sum over the given time period, and so reflect individual customer's behaviors in the corresponding shopping process over multiple shopping visits.

From the above notation, we define the customer profile as the matrix of ratings $\mathbf{P} = (p_{ij})$, $i = 1, \ldots, M$ (total number of customers), $j = 1, \ldots, |G|$ (total number of grain product classes), where

$$p_{ij} = \frac{p_{ij}^c - \min\limits_{1\le j\le |G|}(p_{ij}^c)}{\max\limits_{1\le j\le |G|}(p_{ij}^c) - \min\limits_{1\le j\le |G|}(p_{ij}^c)} + \frac{p_{ij}^b - \min\limits_{1\le j\le |G|}(p_{ij}^b)}{\max\limits_{1\le j\le |G|}(p_{ij}^b) - \min\limits_{1\le j\le |G|}(p_{ij}^b)} + \frac{p_{ij}^p - \min\limits_{1\le j\le |G|}(p_{ij}^p)}{\max\limits_{1\le j\le |G|}(p_{ij}^p) - \min\limits_{1\le j\le |G|}(p_{ij}^p)}. \tag{2}$$

p_{ij} is a sum of the normalized value of p_{ij}^c, p_{ij}^b and p_{ij}^p. It ranges from 0 to 3, where more preferred product results in bigger value. Note that the weights for each shopping step are not the same although they look equal as in Eq. (2). From a casual fact that customers who purchased a specific product had already not only clicked several Web pages related to it but placed it in the shopping basket, we can see that Eq. (2) reflects the different weights.

3.4. Phase 3: neighborhood formation

This phase performs computing the similarity between customers and, based on that, forming a neighborhood between a target customer and a number of like-minded customers. The process follows the same manner as that of typical nearest-neighbor algorithms except forming the neighborhood in reduced dimensional space. The details of the neighborhood formation are as follows.

Given the customer profile $\mathbf{P}$, the similarity between two customers a and b, denoted by sim(a, b), is usually measured using either the *correlation* or the *cosine* measure (Sarwar et al., 2000b).

- *Correlation*. The similarity between two customers a and b is measured by calculating the *Pearson-r* correlation

corr_{ab}, which is given by

$$\mathrm{sim}(a,b)=\mathrm{corr}_{ab}=\frac{\sum_i (p_{ai}-\bar{p}_a)(p_{bi}-\bar{p}_b)}{\sqrt{\sum_i (p_{ai}-\bar{p}_a)^2 \sum_i (p_{bi}-\bar{p}_b)^2}} \tag{3}$$

where p_{ai} and p_{bi} are customer a and b's ratings on product i, respectively, and $\bar{p}_a$ and $\bar{p}_b$ are customer a and b's average ratings on all products, respectively.

- *Cosine*. Two customers a and b are thought of as two vectors in the $|G|$ dimensional product space. The similarity between two customers a and b is measured by calculating the cosine of the angle between the two vectors, $\cos(\vec{a},\vec{b})$, which is given by

$$\mathrm{sim}(a,b)=\cos(\vec{a},\vec{b})=\frac{\mathbf{P}_a\cdot\mathbf{P}_b}{\|\mathbf{P}_a\|\|\mathbf{P}_b\|} = \frac{\sum_{k=1}^{|G|} p_{ak}\times p_{bk}}{\sqrt{\sum_{k=1}^{|G|} p_{ak}^2}\times\sqrt{\sum_{k=1}^{|G|} p_{bk}^2}} \tag{4}$$

where $\mathbf{P}_a$ and $\mathbf{P}_b$ are row vectors of the customer profile $\mathbf{P}$ for two customers a and b, respectively.

WebCF-PT employs the correlation as the similarity measure because previous research (Breese et al., 1998) has shown its superiority in performance over the other.

Using the correlation measure, this phase determines which customers are used in the recommendation for the target customer. Two well-known techniques, *correlation-thresholding* and *best-n-neighbors*, are generally used to determine how many neighbors to select (Herlocker et al., 1999). Correlation-thresholding technique is to form a neighborhood as customers with absolute correlates greater than a given threshold, while best-n-neighbors technique is to select the best n correlates for the neighbors. In WebCF-PT, the best-n-neighbors technique is adopted because of its superiority in performance over the other (Herlocker et al., 1999).

3.5. Phase 4: recommendation generation

The final phase of WebCF-PT is to ultimately derive the *top-N* recommendation from the neighborhood of customers. For each customer, we produce a recommendation list of N products that the target customer is most likely to purchase. Previously purchased products are excluded from the recommendation list in order to broaden each customer's purchase patterns or coverage. We suggest three different methods for generating a recommendation list for a given customer.

- *Recommendation of the most frequently purchased product (MFP)*. This technique, adopted from the study of Sarwar et al. (2000b), looks into the neighborhood and for each neighbor, scans through a sales database and counts the purchase frequency of the products. After all neighbors are accounted for, the system sorts the products according to their frequency count and returns the N most frequently purchased products as the recommendation list. This method assumes that the more a product is purchased, the more popular it becomes.
- *Recommendation of the most frequently referred product (MFR)*. Unlike MFP based on purchase frequencies of all neighbors, this method sorts the products according to their reference frequencies. The reference frequency of the neighborhood of a particular customer a for a product j $(1 \le j \le N)$, $\mathrm{RF}_{a,j}$, is defined below:

$$\mathrm{RF}_{a,j}=\sum_{i\in \text{neighbors of customer } a}\left(\frac{r_{ij}^c-\min\limits_{1\le j\le N}(r_{ij}^c)}{\max\limits_{1\le j\le N}(r_{ij}^c)-\min\limits_{1\le j\le N}(r_{ij}^c)}+\frac{r_{ij}^b-\min\limits_{1\le j\le N}(r_{ij}^b)}{\max\limits_{1\le j\le N}(r_{ij}^b)-\min\limits_{1\le j\le N}(r_{ij}^b)}+\frac{r_{ij}^p-\min\limits_{1\le j\le N}(r_{ij}^p)}{\max\limits_{1\le j\le N}(r_{ij}^p)-\min\limits_{1\le j\le N}(r_{ij}^p)}\right)$$

where N is the number of products, and r_{ij}^c, r_{ij}^b and r_{ij}^p is the total number of occurrences of click-throughs, basket placements and purchases of a customer i for a product j, respectively. This method follows from the hypothesis that the more a product is referred, the higher the possibility of product's purchase becomes. The reference frequency is computed using clickstream data as in building the customer profile.

- *Recommendation of product with the highest click-to-buy rate (HCR)*. Based on customer clickstream patterns like MFR, this method chooses products with the highest click-to-buy rate of all neighbors. The click-to-buy rate indicates how many click-throughs are converted to purchases. Formally, the click-to-buy rate of the neighborhood of a particular customer a for a product j $(1 \le j \le N)$, $\mathrm{CTB}_{a,j}$, is defined below:

$$\mathrm{CTB}_{a,j}=\frac{\sum_{i\in \text{neighbors of customer } a} r_{ij}^p}{\sum_{i\in \text{neighbors of customer } a} r_{ij}^c} \tag{6}$$

where N is the number of products, and r_{ij}^c and r_{ij}^p is the total number of occurrences of click-throughs and purchases of a customer i for a product j, respectively. This method assumes that the higher click-to-buy rate a product has, the more marketing-efficient it becomes (Lee et al., 2001).

The decision of choosing a method is determined by domain specific knowledge and heuristic knowledge of marketing managers. For facilitating such a decision, we evaluate the relative performance of each method by performing an experiment with real-world data.

4. Experimental evaluation

4.1. Data sets

For our experiments, we used Web log data and product data from C Web retailer in Korea that sells a variety of beauty products.

4.1.1. Web log data

One hundred and twenty-four Web log files were collected from four IIS Web servers during the period between May 1 and 30, 2001. The total size of the Web log files was about 64,730 MB, and total number of HTTP requests was about 420,000,000,000. For application to our experiments, data preprocessing tasks such as data cleansing, user identification, session identification, and path completion were applied to the Web log files. Finally, we obtained a transaction database in the form of ⟨time, customer-id, product-id, shopping-step⟩ where the shopping-step is the click-through step, the basket-placement step or the purchase step. This database contains the transactions of 66,329 customers. In total, the database contains 2,249,540 records that consist of 7208 purchase records, 60,892 basket-placement records, and 2,181,440 click-through records. Fig. 7 provides a part of raw Web log data and the corresponding transaction database.

The period between May 1 and 24, 2001 was set as training period and the period between May 25 and 30, 2001 was set as test period. As the target customers, we selected 116 customers who have purchased at least one product in both periods. Finally, the training set consisted of 8960 transaction records created by the target customers for the training period, and the test set consisted of 156 purchase records created by them for the test period.

4.1.2. Product data

C Web retailer deals in 3216 products. The product taxonomy consists of three levels plus the root *All*. The top level (level 1) contains 10 product classes, the next level (level 2) contains 72 product classes, and bottom level (level 3) contains 3216 products.

4.2. Evaluation metrics

The existing studies about recommender systems have used a number of different measures for evaluating the success of a recommender system. The main objective of this research is to develop an effective and efficient recommendation methodology that has better quality and less computation time compared to other methodologies. This research employs two evaluation metrics for evaluating our methodology in terms of quality and performance requirements.

4.2.1. Quality evaluation metric

With the training set and the test set, WebCF-PT works on the training set first, and then it generates a set of recommended products, called recommendation set, for a given customer. To evaluate the quality of the recommendation set, *recall* and *precision* have been widely used in recommender systems research (Basu et al., 1998; Billsus & Pazzani, 1998; Lin et al., 2000, 2002; Sarwar et al., 2000b). Recall is defined as the ratio of the number of products in both the test set and the recommendation set to the number of products in the test set. Precision is defined as the ratio of the number of products in both the test set and the recommendation set to the number of products in the recommendation set. Recall means how many of all the products in the actual customer purchase list are recommended correctly whereas precision means how many of the recommended products belong to the actual customer purchase list. These measures are simple to compute and intuitively appealing, but they are in conflict since increasing the size of the recommendation set leads to an increase in recall but at the same time a decrease in precision (Sarwar et al., 2000b). Hence, a widely used combination metric called *F1 metric* (Billsus & Pazzani, 1998; Kim et al., 2002; Rijsbergen, 1979; Sarwar et al., 2000a,b) that gives equal weight to both recall and precision was employed for our evaluation. It is computed as follows:

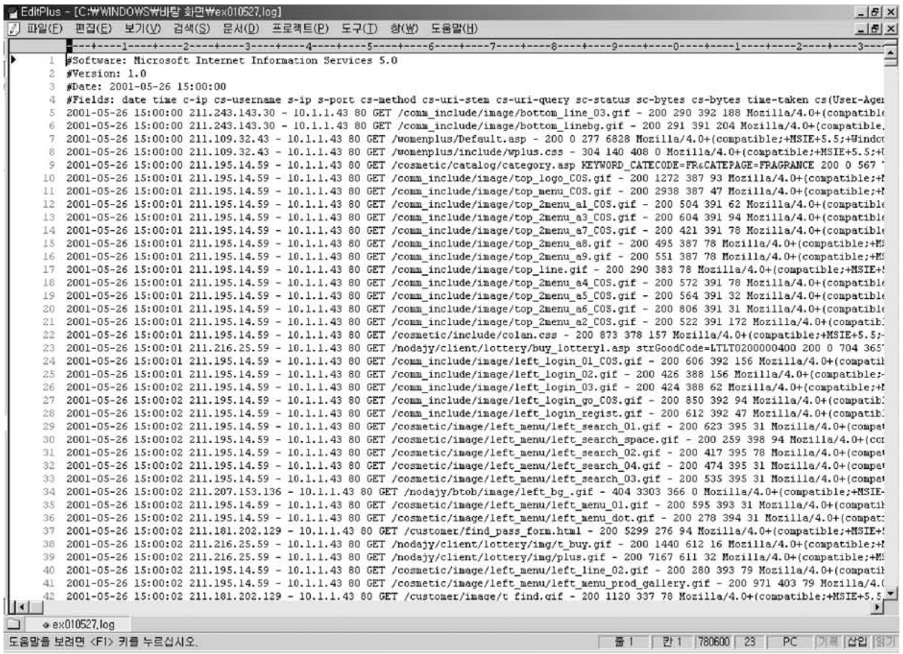
$$F1 = \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}}. \tag{7}$$

4.2.2. Performance evaluation metric

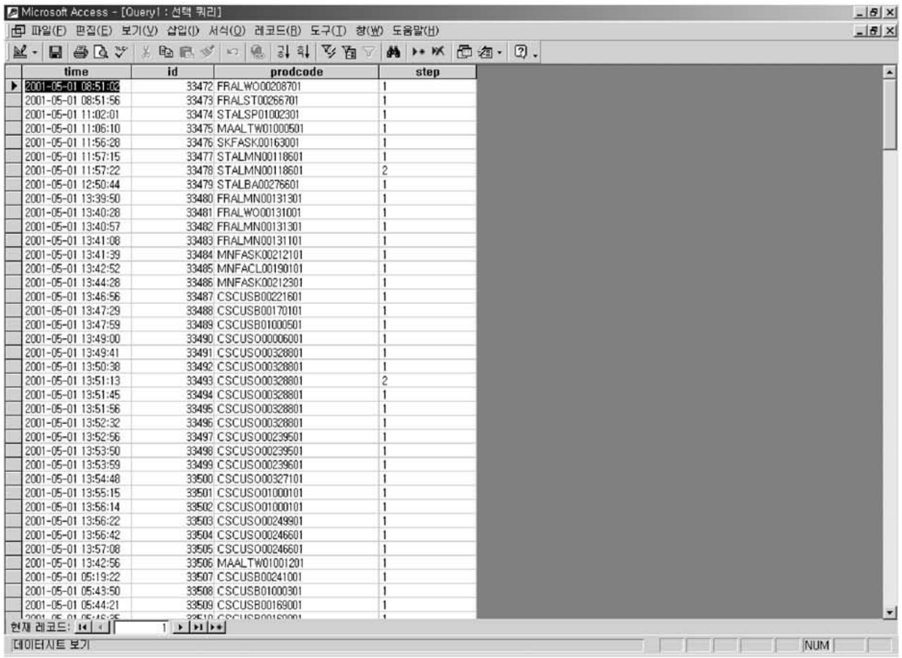
To evaluate the scalability issue, we used a performance evaluation metric in addition to the quality evaluation metric. The *response time* and *throughput* was employed to measure the system performance. The response time defines the amount of time required to compute all the recommendations for the training set, and the throughput denotes the rate at which the recommendations were computed in terms of recommendations per second.

4.2.3. Experimental environments and benchmark recommender systems

A system to perform our experiments was implemented using Visual Basic 6.0 and ADO components. The system consists of two parts—one for Web log preprocessing and the other for experiment execution and result analysis. MS-Access is used to store and process all the data necessary for our experiments. We run our experiments on Windows 2000-based PC with Intel Pentium III processor having a speed 750 MHz and 1 GB of RAM. To verify the effectiveness of WebCF-PT, we compare our results to those of a benchmark recommender system. For this



(a) raw Web log data



(b) transaction database

Fig. 7. Web log preprocessing.

purpose, we developed a typical nearest neighbor CF-based recommender system that has used as a benchmark recommender system in the study of Sarwar et al. (2000a). It was also implemented in Visual Basic 6.0.

4.3. Experiment results and discussions

This section presents a detailed experimental evaluation of the different parameters for the phases of WebCF-PT and compares their performance to those of the benchmark CF algorithm. As the combination of different parameters is enormous, we first determine the optimal values of different parameters, and then use them for the rest of the experiments. Main experimental parameters of WebCF-PT considered during our experiments were as follows:

- Size of neighborhood
- Grain specification

- Method of recommendation generation
- Number of recommended products.

4.3.1. Impact of neighborhood size

The size of the neighborhood is reported to have a significant impact on the recommendation quality (Herlocker et al., 1999; Sarwar et al., 2000b). To determine the sensitivity of neighborhood size, we performed an experiment where we varied the number of neighbors and computed the corresponding *F1* metric. Fig. 8 shows our experimental results. Looking into the results, we can conclude that the size of the neighborhood does affect the quality of *top-N* recommendations. Generally, the recommendation quality increases as the number of neighbors increases, but, after a certain peak, the improvement gains diminish and the quality becomes worse (Sarwar et al., 2000b). Choosing too many neighbors is thought to result in too much noise for those who have high correlates. In our experiment, the peak was reached at the area between 40 and 60. We thus used 50 as our choice of neighborhood size.

4.3.2. Impact of grain specification

In order to evaluate the impact of grain specification on the recommendation quality, we performed an experiment with four types of grains: grain at level 1 (labeled L1), grain at level 2 (labeled L2), level-crossing grain (labeled LC), and grain on adjusted taxonomy (labeled AT) as represented in Figs. 3 and 6. LC was specified by a chief marketing manager of C Web retailer. For specifying AT, we selected 10 level-2 grains of which the dimensionality reduction threshold varies from 20 to 65 in an increment of 5 and computed *F1* for each of them. The average value of *F1* was used. Fig. 9 shows the comparative results obtained from these four grains. Looking into the results, we can see that the recommendation quality of AT is the best and that of LC is the next. This indicates that the grain with an even distribution among products leads to the better quality of recommendations, so we select AT as the grain for the rest of our experiments.

4.3.3. Impact of recommendation generation method

To compare the relative performance of MFP, MFR and HCR method in the recommendation generation, we performed an experiment where we set all of our parameters

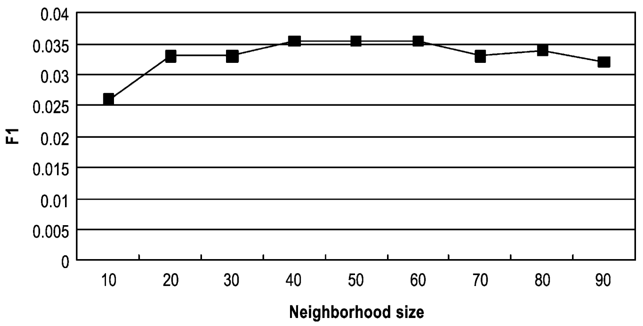


Fig. 8. Impact of neighborhood size.

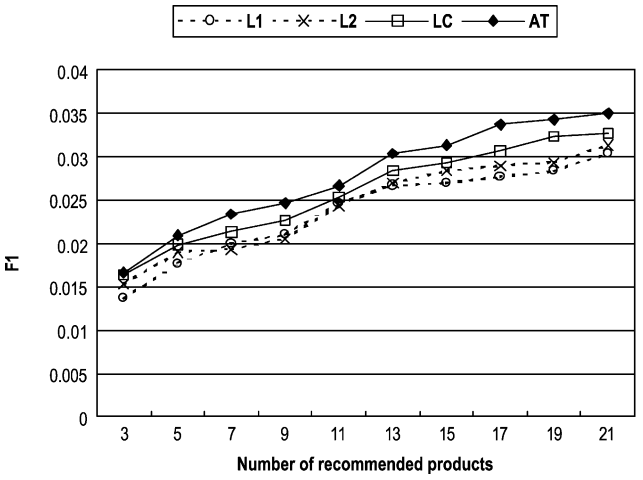


Fig. 9. Impact of grain specification.

to a fixed value, and apply these three different methods on the data sets. Our results are shown in Fig. 10. As we can see from the results, the relative performance differences among these methods are fairly large. MFR works better than the two other methods. This result shows evidence that the usage of Web log data increases the quality of recommendation than using purchase data only. We used MFR as the recommendation generation method for the remaining experiments.

4.3.4. Quality comparison with the benchmark CF algorithm

Once we obtained the optimal values of the parameters, we compared the recommendation quality of WebCF-PT with that of the benchmark CF algorithm. Fig. 11 shows our experimental results. It can be observed from Fig. 11 that WebCF-PT works better than the benchmark CF algorithm at all the number of recommended products. WebCF-PT that uses the optimal choice for each of its parameters works even better, achieving an average improvement of 32%.

4.4. Performance comparison with the benchmark CF algorithm

To compare the performance of WebCF-PT with that of the benchmark CF algorithm, we performed an experiment

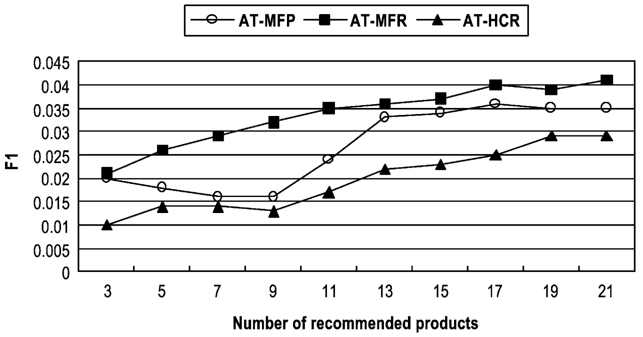


Fig. 10. Impact of recommendation generation method.

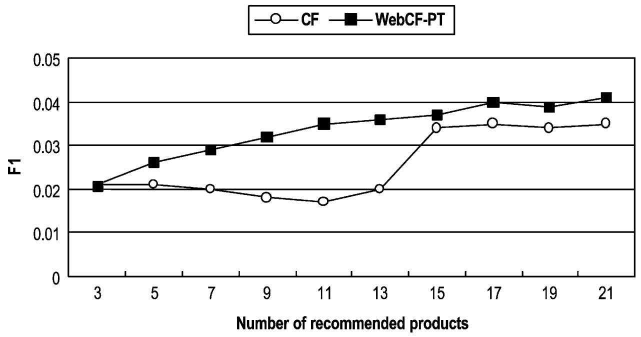


Fig. 11. Quality comparison of WebCF-PT and benchmark CF algorithm.

Table 1
Performance comparison of WebCF-PT and benchmark CF algorithm

	Benchmark CF	WebCF-PT
Response time (s)	91.53	4.87
Throughput	20.00	362.00

to measure the response time and the throughput of each algorithm. Table 1 shows the response time and the throughput provided by the two algorithms. Looking into the results shown in Table 1, we can see that WebCF-PT is about 18 times faster than the benchmark CF algorithm. This point is very important because the number of products and that of customers grow very fast with the widespread use of e-commerce.

5. Conclusion

The rapid expansion of e-commerce forces existing recommender systems to deal with a large number of customers and products and to ensure high quality of recommendations. In this paper, we focused on these challenging issues of the recommender systems and proposed a recommendation methodology where we apply Web usage mining and the product taxonomy to address these issues together. Some experiments show that the methodology works better and faster than existing CF-based recommender systems in an e-commerce environment.

The research work presented in this paper makes several contributions to the recommender systems related research. First, we applied the product taxonomy both to reducing the sparsity in the rating database and to improving the scalability of searching for neighbors. Second, we developed a Web usage mining technique to capture implicit ratings by tracking customers' shopping behaviors on the Web and applied it to reducing the sparsity. Third, we developed a Web usage mining technique to choose proper products to recommend from the neighborhood.

While our experimental results suggest that the proposed methodology is effective and efficient for product recommendations in the Internet business environment, these results are based on data sets limited to the particular e-commerce site that has a small number of customers, products, and transactions. Therefore, it is required to evaluate our methodology in more detail using data sets from a variety of large e-commerce sites. Furthermore, it will be an interesting research area to conduct a real marketing campaign to target customers using our methodology and then to evaluate its performance.

References

Adomavicius, G., & Tuzhilin, A. (2001). Expert-driven validation of rule-based user models in personalization applications. *Data Mining and Knowledge Discovery*, *5*(1–2), 33–58.

Aggarwal, C. C., & Yu, P. S. (2000). Data mining techniques for personalization. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, *23*, 4–9.

Balabanović, M., & Shoham, Y. (1997). Fab: Content-based, collaborative recommendation. *Communications of the ACM*, *40*(3), 66–72.

Basu, C., Hirsh, H., & Cohen, W. (1998). Recommendation as classification: Using social and content-based information in recommendation. *Proceedings of the 15th National Conference on Artificial Intelligence*, 714–720.

Berry, J. A., & Linoff, G. (1997). *Data mining techniques: For marketing, sales, and customer support*. New York: Wiley.

Billsus, D., & Pazzani, M. J. (1998). Learning collaborative information filters. *Proceedings of the 15th International Conference on Machine Learning*, 46–54.

Breese, J. S., Heckerman, D., & Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, 43–52.

Cho, Y. H., Kim, J. K., & Kim, S. H. (2002). A personalized recommender system based on Web usage mining and decision tree induction. *Expert Systems with Applications*, *23*(3), 329–342.

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., & Sartin, M. (1999). Combining content-based and collaborative filters in an online newspaper. *Proceedings of the ACM SIGIR'99 Workshop on Recommender Systems, Berkeley, CA*, *August*.

Condliff, M., Lewis, D. D., Madigan, D., & Posse, C. (1999). Bayesian mixed-effects models for recommender systems. *Proceedings of the ACM SIGIR'99 Workshop on Recommender Systems, Berkeley, CA*, *August*.

Cooley, R., Mobasher, B., & Srivastava, J. (1999). Data preparation for mining world wide web browsing patterns. *Journal of Knowledge and Information Systems*, *1*, 5–32.

Good, N., Schafer, J. B., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J., & Riedl, J. (1999). Combining collaborative filtering with personal agents for better recommendations. *Proceedings of the AAAI'99*, 439–446.

Han, J., & Fu, Y. (1994). Dynamic generation and refinement of concept hierarchies for knowledge discovery in databases. *Proceedings of the AAAI'94 Workshop on Knowledge Discovery in Databases*, 157–168.

Han, J., & Fu, Y. (1995). Discovery of multiple-level association rules from large databases. *Proceedings of the International Conference on Very Large Data Bases*, 420–431.

Han, J., & Fu, Y. (1999). Mining multiple-level association rules in large databases. *IEEE Transactions on Knowledge and Data Engineering*, *11*(5), 798–804.

Herlocker, J., Konstan, J., Borchers, A., & Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. *Proceedings of the Conference on Research and Development in Information Retrieval*, 230–237.

Hill, W., Stead, L., Rosenstein, M., & Furnas, G. (1995). Recommending and evaluating choices in a virtual community of use. *Proceedings of the ACM Conference on Human Factors in Computing Systems*, 194–201.

Hoffman, T., & Puzicha, J. (1999). Latent class models for collaborative filtering. *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, 688–693.

Kim, J. K., Cho, Y. H., Kim, W. J., Kim, J. R., & Suh, J. Y. (2002). A personalized recommendation procedure for Internet shopping support. *Electronic Commerce Research and Applications*, *1*(3), 301–313.

Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R., & Riedl, J. (1997). GroupLens: Applying collaborative filtering to usenet news. *Communications of the ACM*, *40*(3), 77–87.

Lawrence, R. D., Almasi, G. S., Kotlyar, V., Viveros, M. S., & Duri, S. S. (2001). Personalization of supermarket product recommendations. *Data Mining and Knowledge Discovery*, *5*(1–2), 11–32.

Lee, J., Podlaseck, M., Schonberg, E., & Hoch, R. (2001). Visualization and analysis of clickstream data of online stores for understanding Web merchandising. *Data Mining and Knowledge Discovery*, *5*(1–2), 59–84.

Lin, W., Alvarez, S. A., & Ruiz, C. (2000). Collaborative recommendation via adaptive association rule mining. *Proceedings of the WEBKDD*.

Lin, W., Alvarez, S. A., & Ruiz, C. (2002). Efficient adaptive-support association rule mining for recommender systems. *Data Mining and Knowledge Discovery*, *6*, 83–105.

Melville, P., Mooney, R. J., & Nagarajan, R. (2001). Content-boosted collaborative filtering. *ACM SIGIR Workshop on Recommender Systems*.

Menascé, D. A., Almeida, V. A., Fonseca, R., & Mendes, M. A. (1999). A methodology for workload characterization of e-commerce sites. *Proceedings of the ACM E-Commerce*, 119–128.

Mobasher, B., Cooley, R., & Srivastava, J. (2000a). Automatic personalization based on Web usage mining. *Communications of the ACM*, *43*(8), 142–151.

Mobasher, B., Dai, H., Luo, T., Sun, Y., & Zhu, J. (2000b). Integrating Web usage and content mining for more effective personalization. *Proceedings of the EC-Web 2000*, 165–176.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An open architecture for collaborative filtering of netnews. *Proceedings of the ACM 1994 Conference on Computer Supported Cooperative Work*, *1994*, 175–186.

Rijsbergen, C. J. (1979). *Information retrieval* (2nd ed.). London: Butterworths.

Rucker, J., & Polano, M. J. (1997). Siteseer: Personalized navigation for the Web. *Communications of the ACM*, *40*(3), 73–75.

Sarwar, B (2001). *Sparsity, scalability, and distribution in recommender systems*. PhD thesis, University of Minnesota.

Sarwar, B., Karypis, G., Konstan, J. A., & Riedl, J. (2000a). Application of dimensionality reduction in recommender system—A case study. *Proceedings of the ACM WebKDD-2000 Workshop*.

Sarwar, B., Karypis, G., Konstan, J. A., & Riedl, J. (2000b). Analysis of recommendation algorithms for e-commerce. *Proceedings of the ACM E-Commerce*, 158–167.

Sarwar, B., Karypis, G., Konstan, J. A., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithm. *Proceedings of the 10th International World Wide Web Conference*, 285–295.

Sarwar, B., Konstan, J., Borchers, A., Herlocker, J., Miller, B., & Riedl, J. (1998). Using filtering agents to improve prediction quality in the GroupLens research collaborative filtering system. *Proceedings of the CSCW'98*, 345–354.

Shardanand, U., & Maes, P. (1995). Social information filtering: Algorithms for automating 'word of mouth'. *Proceedings of the Conference on Human factors in Computing Systems*, 210–217.

Soboroff, I. M., & Nicholas, C. K. (1999). Combining content and collaboration in text filtering. *Proceedings of the IJCA'99 Workshop on Machine Learning in Information Filtering*, *'99*, 86–91.

Srivastava, J., Cooley, R., Deshpande, M., & Tan, P. (2000). Web usage mining: Discovery and applications of usage patterns from Web data. *SIGKDD Explorations*, *1*(2), 1–12.

Terveen, L., Hill, W., Amento, B., McDonald, D. W., & Creter, J. (1997). PHOAKS: A system for sharing recommendations. *Communications of the ACM*, *40*(3), 59–62.

Ungar, L. H., & Foster, D. P. (1998). A formal statistical approach to collaborative filtering. *Proceedings of the Conference on Automated Learning and Discovery*.

VanderMeer, D., Dutta, K., & Datta, A. (2000). Enabling scalable online personalization on the Web. *Proceedings of the ACM E-Commerce*, 185–196.