

# Modeling Evolutionary Behaviors for Community-based Dynamic Recommendation

Xiaodan Song, Ching-Yung Lin  
Department of Electrical Engineering,  
University of Washington, Box  
352500, Seattle, WA 98195 USA  
{song, cylin}@ee.washington.edu

Belle L. Tseng  
NEC Laboratories America  
10080 N. Wolfe Rd, SW3-350  
Cupertino, CA 95014 USA  
belle@sv.nec-labs.com

Ming-Ting Sun  
Department of Electrical Engineering,  
University of Washington, Box 352500,  
Seattle, WA 98195 USA  
sun@ee.washington.edu

## Abstract

We exploit dynamic patterns from both documents' and users' aspects to build models for recommendation. We propose a Community-Based Dynamic Recommendation (CBDR) scheme to make recommendations by taking content semantics, evolutionary patterns, and user communities into consideration. A Time-Sensitive Adaboost algorithm is proposed to build adaptive user models for ranking document candidates based on leveraging dynamic factors such as freshness, popularity, and other attributes. Our experimental results on a large online application system demonstrate the recommendation usefulness of the CBDR scheme is 259% better than the collaborative filtering, 126% better than the community-based static recommendation algorithm, and 106% better than the optimal global recommendation bound.

**Keywords:** Dynamic recommendation, collaborative filtering, content based filtering, adaptive user modeling

## 1. Introduction

Recommender systems help users deal with information overload by providing personalized recommendations. "Will user  $U$  like item  $X$ ?" is a basic question in the personalized recommendation. *Content-based filtering*, *collaborative filtering*, and *hybrid approaches* are three major categories of methods to answer this question [1]. Content-based filtering is to characterize what a user likes, based on the past history of the user activity and the classification of the items [2]. They are usually limited by the features that are explicitly associated with the items [1]. Collaborative filtering infers an individual's interests/preferences from that of the other people with similar tastes [3-6]. The fact that new items must be chosen by substantial users before they can be recommended to other users creates what is known as the 'cold start' problem [6]. Hybrid approaches by combining collaborative and content-based methods have been proposed to help avoiding the above mentioned limitations of content-based and collaborative systems [6-8].

Both the importance of items and the interests of users are naturally dynamic. Related works on

exploring dynamic facts include [9-11]. Using content-based approaches, [9] studies how to learn users' long-term and short-term interest categories in a dynamic environment. They propose a model to represent a user's interests, which maintains a long-term interest descriptor to capture the user's general interests and two short-term interest descriptors (a positive and a negative) to keep track of the user's more recent, faster-changing interests. [10] presents a multidimensional approach to present the temporal and other contextual information besides the user and item-rating information. A reduction based approach is proposed to select the related information so that the multidimensional data are reduced to the traditional two-dimensional User  $\times$  Item recommendation space, then traditional collaborative filtering can be applied. [11] adopts forgetting weights to model evolving user interests in clickstreams by taking into account the age of each user activity/session.

Compared to other recommendation systems, the specific problem we address in this paper is how a system can adapt to the dynamic aspects of both the value of content and the evolution of user interests. The Value of items may change over time. For instance, some documents are time-sensitive and expire quickly with a life-time as short as several days, such as news. Some documents are of continuous interests for many people, such as classic technical papers. People usually have different intentions. Some users are more interested in breaking information, so they mostly read news. Others tend to look for long-term documents. Thus, content semantic analysis of items is necessary. Another important point is that people's interests also evolve over time. One person, who was interested in a certain topic one year ago, may not care about the same topic any more and changes the interests to something else. Mutual influence in a community is also a deciding factor of people's interests. Thus, we consider *content semantics*, *evolutionary patterns*, and *user communities* as three key analytic components for dynamic recommendations. The main contribution of this paper is providing a dynamic user modeling method - Community-Based Dynamic Recommendation (CBDR) scheme, to capture the contextual and dynamic patterns of users' interests

based on the past history of content semantics, evolutionary patterns, and user communities simultaneously. We propose a Time-Sensitive Adaboost algorithm to build adaptive user models that will be used for ranking documents in a community-based recommendation system.

The rest of the paper is organized as follows. In Section 2, we present the CBDR system. In Section 3, we address the problem of how to leverage dynamic patterns from both documents' and users' aspects. In Section 4, a Time-Sensitive Adaboost algorithm is proposed and described in details. In Section 5, we propose an evaluation metric, and show experimental results to demonstrate the performance of our CBDR scheme. Conclusion and future works are addressed in Section 6.

## 2. System Description

Without loss of generality, we describe our proposed scheme using a dataset collected from NEC's "EigyoRyoku 21" (denoted as ER and stands for Sales-Force in Japanese) system. The ER system is a knowledgebase to support the sales staff with registered documents, that include articles, slides, *etc.*. We collected a one-year period of clickstream log files from Apr. 2004 to Mar. 2005 covering over 30,000 users and over 20,000 documents. Users' demographic information, and the authors, titles, abstracts, and disclosure timestamps of the documents are included in the dataset. The clickstream log is partitioned into sessions that start with "login", and then represent a collection of ordered sequences of user actions. The documents in the system are heterogeneous, i.e., they include short-term documents such as announcements, and long-term documents such as technical manuals. In this paper, we assume if a user downloads one document, he or she is interested in it.

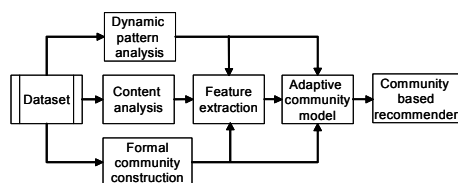


Figure 1. An overview of our proposed CBDR scheme

Figure 1 illustrates the overall structure of our proposed CBDR scheme. In the community construction step, formal communities are constructed based on the organization chart if available, which is simply based on the fact that people from the same department of the same company tend to have similar interests, since they tend to have similar background and are working on similar projects. In the content analysis step, Latent Dirichlet Allocation (LDA) [12] is conducted to recognize the topics of documents from the titles and abstracts of the documents. In the

dynamic pattern analysis step, dynamic patterns from both users and documents aspects are tackled. According to how long documents are being accessed, they are classified into two types: short-term and long-term. Then, users' intentions of updating news or looking for long-term documents are classified based on the types of documents they download. Furthermore, we predict the expiration dates of documents in order to suitably retire them from the recommendation candidate list. The details of dynamic pattern analysis will be further addressed in Section 3.

Instead of relying only on content or collaborative aspect from what users like, our algorithm gains insight into users' preferences on why they like these items from both static and dynamic aspects. The static features include topics of documents extracted from content analysis step, collaborative aspect, in which user's formal community is extracted from formal community construction step. The dynamic features include the freshness of the documents, short-term or long-term type of the document, popularity of the documents, and user intention.

After content analysis, dynamic pattern analysis, formal community construction and feature extraction, in the adaptive community modeling step, we propose a Time-Sensitive Adaboost algorithm to adaptively model users' evolving interests and preference for ranking documents in a community-based recommendation system, which will be described in details in Section 4.

## 3. Analysis of Dynamic Patterns

In order to provide dynamic recommendations, we need to explore what dynamic patterns exist in both documents and user's behaviors, and understand how they affect the recommendations.

### 3.1 From documents

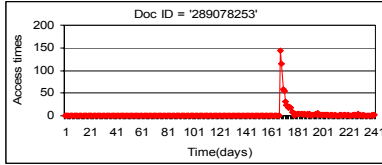
#### 3.1.1 Document access patterns

We define the life-time of a document as the time period when 90% of the downloading behavior occurred to estimate the expected useful life-time of the documents in the ER database. First of all, all weekends and holidays are removed from the database. In total, 241 days were left as the valid working days. Then, users' self-downloading and administrators' downloading behaviors are regarded as noise and also removed from the statistics.

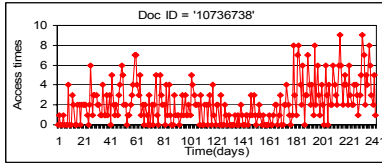
Based on our study, we found two types of downloading patterns exist in the dataset. The life-time histograms for two representative documents are illustrated in Figure 2. Figure 2(a) shows the pattern of a document with short-term interest. A spike in the access histogram is shown on the first day when this document is disclosed, and then quickly drops down to zero, and nobody downloads it any more after 15 working days. For the other type shown in Figure 2(b),

people demonstrated continuous interest on this document during the whole year.

By applying a Gaussian Mixture Model, documents are clustered into two groups with the average life-time as 34.5 days and 146.9 days respectively. 54% of the documents belong to short-term, and 46% of them belong to long-term. By manual verification, we found that documents belonging to the short-term type are mostly time-sensitive, e.g., announcements, conference notes, and executive summaries. Documents belonging to the long-term type are mostly of continuing interests, such as handouts, specifications, and reference documents. Generally, short-term documents are related to news, which are interesting at the time they are discovered, but not any more after a short time since it has become a fact. On the other hand, technical documents usually have long time interests for users.



(a) Document with peak on one day followed by a decay



(b) Document of sustained interest

**Figure 2.** Document downloading patterns: user access histogram (y-axis) with respect to the day on which people accessed (x-axis)

### 3.1.2 Document expiration date prediction

Based on our observation, we found that the longer nobody downloads a document, the less likely it will be downloaded in the future. Thus, the probability of one document being expired is modeled as:

$$P(e) = 1 - \exp(-\eta \Delta t) \quad (1)$$

where

$$\Delta t = \begin{cases} t_{na} - thr_t & \text{if } t_{na} > thr_t \\ 0 & \text{others} \end{cases}$$

$t_{na}$  measures how many days one document has not been downloaded,  $\eta$  is a parameter to control the expiration rate, and  $thr_t$  is a sensitive threshold. Based on our experiments, we found  $\eta = 0.25$  and  $thr_t = 15$  are good choices, the results are not sensitive to  $\eta$ , and the value between 14 and 28 is good for  $thr_t$ .

### 3.2 From users

Users' intention of using the system can be

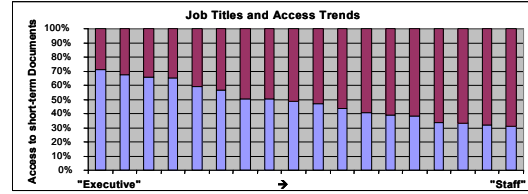
different: some of them go there to update information, so they only read what's new; some of them go there looking for informative long-term materials. If we can differentiate users' intention based on the historical data, recommendations can be improved by fitting their needs of short-term or long-term documents.

Based on the types of documents we investigated in Section 3.1, we collect the types of the documents a user downloaded, and estimate the probability of how much does one user like short-term documents by

$$P(D_s | u) = \frac{\#(\text{documents in short-term which the user downloaded})}{\#(\text{documents which the user downloaded})} \quad (2)$$

where  $u$  represents one user,  $D_s$  represents short-term documents.

In Figure 3, we study the correlation of the users' accessed document types (short-term or long-term) and their job titles. Interestingly, we found that higher-level managers tend to access short-term documents, while other employees tend to access long-term documents, which verifies the possibility of inferring users' intention from the documents they accessed.



**Figure 3.** The correlation of job titles and access types

Users' interests and preferences also evolve over time. A user's recent preferences are more important than the preferences from a long time ago with respect to the current needs. The Time-Sensitive Adaboost model as in Section 4 can be used for modeling this.

## 4. Time-Sensitive Adaboost

In this section we propose a Time-Sensitive Adaboost algorithm to adapt users' evolving interests.

Adaboost, an algorithm for constructing a "strong" hypothesis as a linear combination of "weak" hypotheses [14], is a feature selector with a principled strategy - minimization of upper bound on empirical error, thus has good generalization properties. Among the state-of-the-art classification algorithms, Adaboost has been used in text categorization, information extraction and filtering, and evaluated as one of the most accurate classification algorithms [14].

Given features  $x$  representing one document,  $P(c=1|x,t)$  is the probability indicating whether a user is interested in this document ( $c=1$ ) during time period  $t$ . We model it as a logistic regression function:

$$P(c=1|x,t) = \frac{\exp(Q(x,t))}{1 + \exp(Q(x,t))} \quad (3)$$

which can also be written as:

$$\log \frac{P(c=1|x,t)}{P(c=-1|x,t)} = Q(x,t) \quad (4)$$

In [13], the authors show that Adaboost algorithm is actually a stage-wise estimation procedure for fitting an additive logistic regression model by minimizing a loss function. The simple rules are trained sequentially; each rule is trained on examples which are most difficult to classify by the preceding rules. They proposed a Real Adaboost procedure which finds an  $F(x,t)$  that minimizes  $E\{\exp(-cF(x,t))\}$  at:

$$F(x,t) = \frac{1}{2} \log \frac{P(c=1|x,t)}{P(c=-1|x,t)} \quad (5)$$

Comparing Eq. (4) and Eq. (5), we obtain  $Q = 2 \cdot F$ . Therefore, plug it into Eq. (3) we obtain:

$$P(c=1|x,t) = \frac{\exp(2 \cdot F(x,t))}{1 + \exp(2 \cdot F(x,t))} \quad (6)$$

Observing more recent behaviors are more useful for predicting the user's current behaviors, we propose a Time-Sensitive Adaboost model to estimate  $F$ . Recall that for Adaboost, our goal is to minimize an exponential loss function:

$$\sum_{i=1}^N \exp\left(-c_i \sum_{s=1}^S \alpha_s h_s(x_i)\right) \quad (7)$$

where  $h_s$  are weak hypotheses,  $\alpha_s$  are parameters to indicate how reliable and important each hypothesis is and  $S$  is the number of hypotheses selected. All the samples are regarded equally important at the beginning of the learning process. For our Time-Sensitive model, the current data are given larger weights indicating the more importance of being separated. Therefore, we change the loss function to:

$$\sum_{i=1}^N \exp\left(-c_i \sum_{s=1}^S \alpha_s \exp(-\tau \cdot (t-t_i)) h_s(x_i, t)\right) \quad (8)$$

where  $t_i$  indicates the timestamp for  $i$ th sample and  $\tau$  is a forgetting parameter which controls how quickly the previous data are forgotten. The goal is achieved by assigning different initial weights to different samples at the beginning of the leaning process. In this paper,  $h_s$  are defined as simple linear classifiers, basically telling whether each feature is more or less than a threshold. The threshold is learned by minimizing the weighted error rate  $\varepsilon_s$  in iteration  $s$ . The details of the algorithm are illustrated in Figure 4.

Finally, after the Time-Sensitive Adaboost algorithm, we obtain:

$$F(x,t) = \sum_{s=1}^S \alpha_s h_s(x,t) \quad (9)$$

Therefore, we plug Eq. (9) into Eq. (6) to estimate  $P(c=1|x,t)$  as:

$$P(c=1|x,t) = \frac{\exp\left(2 \cdot \sum_{s=1}^S \alpha_s h_s\right)}{1 + \exp\left(2 \cdot \sum_{s=1}^S \alpha_s h_s\right)} \quad (10)$$

**Algorithm: Time-Sensitive Adaboost**

**Given:**  $(x_1, c_1, t_1), \dots, (x_N, c_N, t_N)$  where  $x_i \in \mathcal{X}$ ,  $c_i \subseteq \{-1, 1\}$ ,  $N$  is the size of samples in the training set; current time  $t$ , and  $\tau$

**For**  $s = 1, \dots, S$

Initialize  $D_1(i) = 1 / (N \cdot \exp(\tau \cdot (t - t_i)))$ .

Set the weight  $\alpha_s$  of the current weak hypothesis

$h_s$  according to its weighted error rate  $\varepsilon_s$

$$\alpha_s = \frac{1}{2} \ln \left( \frac{1 - \varepsilon_s}{\varepsilon_s} \right)$$

where  $\varepsilon_s = \sum_{i=1}^N D_s(i) h_s(x_i) c_i$ .

Update  $D_{s+1}(i) = \frac{D_s(i) \exp(-\alpha_s c_i h_s(x_i))}{Z_s}$

where  $Z_s$  is a normalization factor.

**End**

Find weak hypothesis by:  $h_s = \arg \min_{h_j} \varepsilon_j$ .

**Output:** the final hypothesis:  $H(x) = \text{sign}(F(x))$

where  $F(x) = \sum_{s=1}^S \alpha_s h_s(x)$ .

**Figure 4.** The Time-Sensitive Adaboost Algorithm

## 5. Experiments

In this section, we first propose a metric for evaluation of users' satisfaction about the recommendations. Then, we demonstrate the performance of our CBDR algorithm.

### 5.1 Evaluation metric

The dataset is divided into two sets: the data from Apr. 2004 to Feb. 2005 serve as the training data, and the data from Mar. 2005 serve as the testing data. For our evaluation, we recommend five documents for each user who logs into the ER system in Mar. 2005. Using the proposed CBDR algorithm, the same list of documents is recommended to people in the same community, where each person is provided five documents that he or she has never downloaded. Our goal is to measure whether the recommended documents match the documents that the user really downloaded.

*Community-based recommendation usefulness metric* is defined as how many people downloads at least one, two, three, four, or five documents from the recommendation list during the testing period. *Global upper bound* is the best performance a system can achieve in terms of *global recommendation*, in which all users will obtain the same recommendations, if it knows which items the users will download during the

