# Unified Relevance Models for Rating Prediction in Collaborative Filtering

JUN WANG
University College London
ARJEN P. DE VRIES
Delft University of Technology and CWI
and
MARCEL J.T. REINDERS
Delft University of Technology

Collaborative filtering aims at predicting a user's interest for a given item based on a collection of user profiles. This paper views collaborative filtering as a problem highly related to information retrieval, drawing an analogy between the concepts of users and items in recommender systems and queries and documents in text retrieval.

We present a probabilistic user-to-item relevance framework that introduces the concept of relevance into the related problem of collaborative filtering. Three different models are derived, namely, a *user-based*, an *item-based* and a *unified relevance model*, estimating their rating predictions from three sources: the user's own ratings for different items, other users' ratings for the same item, and, ratings from different but similar users for other but similar items.

To reduce the data sparsity encountered when estimating the probability density function of the relevance variable, we apply the non-parametric (data-driven) density estimation technique known as the *Parzen-window method* (or, kernel-based density estimation). Using a Gaussian window function, the similarity between users and/or items would however be based on Euclidean distance. Because collaborative filtering literature has reported improved prediction accuracy when using cosine similarity, we generalise the Parzen-window method by introducing a *projection kernel*.

Existing user-based and item-based approaches correspond to two simplified instantiations of our framework. User-based and item-based collaborative filtering represent only a partial view of the prediction problem, where the unified relevance model brings these partial views together under the same umbrella. Experimental results complement the theoretical insights with improved recommendation accuracy. The unified model is more robust to data sparsity, because the different types of ratings are used in concert.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval - *Information Filtering*

Additional Key Words and Phrases: Collaborative Filtering, Recommendation, Personalisation

Authors' addresses: J. Wang is with Adastral Park Campus, University College London, UK, E-mail: j.wang@adastral.ucl.ac.uk; Arjen de Vries is with CWI, the Netherlands, E-mail: arjen@acm.org; M.J.T. Reinders is with the Department of Mediamatics, Delft University of Technology, the Netherlands, E-mail: j.t.reinders@tudelft.nl.
The work was done when the first author was with the Department of Mediamatics, Delft University of Technology.

## 1. INTRODUCTION

Collaborative filtering (CF) algorithms use a collection of user profiles to identify interesting "information" (items) for these users. These profiles result from asking users explicitly to rate items (rating-based CF), or, they are inferred from log-archives (log-based CF) [Hofmann 2004; Wang et al. 2006a]. The profiles can be thought of as the evidence (observation) of *relevance* between users and items. Thus, the task of collaborative filtering is to predict the unknown relevance between the test user and the test item [Wang et al. 2006a].

Relevance is also a very important concept in the text retrieval domain and has been heavily studied (e.g., [van Rijsbergen 1979; Lavrenko 2004]). Many probabilistic approaches have been developed to model the estimation of relevance, ranging from the traditional probabilistic models [Robertson and SparckJones 1976] to the latest developments on language models of information retrieval [Lavrenko and Croft 2001; Lafferty and Zhai 2003].

Despite the concept of relevance existing in both collaborative filtering and text retrieval, collaborative filtering has often been formulated as a self-contained problem, apart from the classic information retrieval problem (i.e. ad hoc text retrieval). Research started with memory-based approaches to collaborative filtering, that can be divided in user-based approaches like [Resnick et al. 1994; Breese et al. 1998; Herlocker et al. 1999; Jin et al. 2004] and item-based approaches like [Sarwar et al. 2001; Deshpande and Karypis 2004]. Given an unknown test rating (of a test item by a test user) to be estimated, memory-based collaborative filtering first measures similarities between the test user and all other users (user-based), or, between the test item and all other items (item-based). Then, the unknown rating is predicted by averaging the (weighted) known ratings of the test item by the similar users (user-based), or the (weighted) known ratings of the similar items by the test user (item-based).

The two approaches share the same drawback as the *document-oriented* and *query-oriented* views in information retrieval (IR) [Robertson 2003]: neither presents a complete view on the problem. In both the user-based and item-based approaches, only partial information from the data embedded in the *user-item matrix* is employed to predict unknown ratings, i.e. using either correlation between user data or correlation between item data. Because of the sparsity of user profile data, however, many ratings will not be available. Therefore, it is desirable to unify the ratings from both similar users and similar items, to reduce the dependency on data that is often missing. Even ratings made by other but similar users on other but similar items can be used to make predictions [Wang et al. 2006b]. Not using such ratings causes the *data sparsity problem* of memory-based approaches to collaborative filtering: for many users and items, no reliable recommendation can be made because of a lack of similar ratings.

Thus it is of great interest to see how we can follow the school of thinking in IR relevance models to solve the collaborative filtering problem, once we draw an analogy between the concept of user and items in collaborative filtering and query and documents in IR. This paper applies IR relevance models at a conceptual level, setting up a unified probabilistic relevance framework to exploit more of the data available in the user-item matrix. We establish the models of relevance by

considering user ratings of items as observations of the relevance between users and items. Different from any other IR relevance models, the densities of the our relevance models are estimated by applying the Parzen-window approach [Duda et al. 2001]. This approach reduces the data sparsity encountered when estimating the densities, providing a data-driven solution, i.e., without specifying the model structure a priori. We then extend the Parzen-window density estimation into a projected space, to provide a generalised distance measure which naturally includes most of commonly used similarity measures into our framework.

We derive three types of models from the framework, namely the *user-based relevance model*, the *item-based relevance model* and the *unified relevance model*. The former two models represent a partial view of the problem. In the unified relevance model, each individual rating in the user-item matrix may influence the prediction for the unknown test rating (of a test item from a test user). The overall prediction is made by averaging the individual ratings weighted by their contribution. The weighting is controlled by three factors: the shape of the Parzen window, the two (user and item) bandwidth parameters, and the distance metric to the test rating. The more a rating contributes towards the test rating, the higher the weight assigned to that rating to make the prediction. Under the framework, the item-based and user-based approaches are two special cases and they are systematically combined. By doing this, our approach allows us to take advantage of user similarity *and* item similarity embedded in the user-item matrix to improve probability estimation and counter the problem of data sparsity.

The remainder of the paper is organized as follows. We first summarise related work, introduce notation, and present additional background information for the three main memory-based approaches, i.e., user-based, item-based and the combined collaborative filtering approaches. We then introduce our unified relevance prediction models. We provide an empirical evaluation of the relationship between data sparsity and the different models resulting from our framework, and finally conclude our work.

## 2. RELATED WORK

### 2.1 Collaborative Filtering

Collaborative filtering approaches are often classified as memory-based or model-based. In the memory-based approach, all rating examples are stored *as-is* into memory (in contrast to learning an abstraction). In the prediction phase, similar users or items are sorted based on the memorized ratings. Based on the ratings of these similar users or items, a recommendation for the test user can be generated. Different views of the correlations embedded in the user-item matrix lead to two different types of approaches: user-based methods (looking at user correlation) [Resnick et al. 1994; Breese et al. 1998; Herlocker et al. 1999; Jin et al. 2004] and item-based methods (looking at item correlation) [Sarwar et al. 2001; Deshpande and Karypis 2004]. These approaches form a heuristic implementation of the "Word of Mouth" phenomenon and are widely used in practice, e.g., [Herlocker et al. 1999; Linden et al. 2003]. The advantage of the memory-based methods over their model-based alternatives is that less parameters have to be tuned; however, the data sparsity problem is not handled in a principled manner. Lately, researchers

have introduced dimensionality reduction techniques to address the data sparsity [Sarwar et al. 2000; Goldberg et al. 2001; Rennie and Srebro 2005]. But, as pointed out in [Huang et al. 2004; Xue et al. 2005], some useful information may be discarded during the reduction. [Xue et al. 2005] clusters the user data and applies intra-cluster smoothing to reduce sparsity. [Hu and Lu 2006] extends this idea, by further adding a linear interpolation between the user-based and item-based approaches within the user cluster. The *similarity fusion* method presented in [Wang et al. 2006b] can be also regarded as an effort along this direction, which implicitly clusters users and items simultaneously. Different from [Hu and Lu 2006], [Wang et al. 2006b] takes a rather formal view on the fusion problem and proposes a multi-layer linear smoothing model, showing that additional ratings from similar users towards similar items are also valuable to counter the data sparsity.

In the model-based approach, training examples are used to generate a model that is able to predict the ratings for items that a test user has not rated before. In this regard, many probabilistic models have been proposed. For example, to consider user correlation, [Pennock et al. 2000] proposed a method called personality diagnosis (PD), treating each user as a separate cluster and assuming a Gaussian noise applied to all ratings. It computes the probability that a test user is of the same "personality type" as other users and, in turn, the probability of his or her rating to a test item can be predicted. On the other hand, to model item correlation, [Breese et al. 1998] utilizes a Bayesian Network model, in which the conditional probabilities between items are maintained. Some researchers have tried mixture models, explicitly assuming some hidden variables embedded in the rating data. Examples include the aspect model [Hofmann 2004; Si and Jin 2003], the cluster model [Breese et al. 1998] and the latent factor model [Canny 2002]. These methods require some assumptions about the underlying data structures and the resulting 'compact' models solve the data sparsity problem to a certain extent. However, the need to tune an often significant number of parameters has prevented these methods from practical usage.

In contrast, the work presented in this paper takes a data-driven approach without assuming any data structure a priori, thus getting rid of the significant number of model parameters. It extends the ideas in [Wang et al. 2006b] by further exploring the usage of the probabilistic model of text retrieval. The Parzen-window method is adopted for probability estimation, causing a natural integration of user and item correlations. As a result, the proposed prediction framework is of a general natural. We shall see how some of the previously proposed methods, such as the PD algorithm [Pennock et al. 2000], the Similarity Fusion method [Wang et al. 2006b] and the linear combination method [Hu and Lu 2006], are equivalent to one of the simplified instantiations of our framework (Table III).

### 2.2 Probabilistic Models for Information Retrieval

Probabilistic (relevance) models for information (text) retrieval have been proposed and tested over decades. Rather than giving a comprehensive overview, here we mainly review the models that are related to the problem of collaborative filtering.

The two different *document-oriented* and *query-oriented* views on how to assign a probability of relevance of a document to a user need result in two different types of practical models [Robertson et al. 1982; Bodoff and Robertson 2004]. The

RSJ probabilistic model of information retrieval [Robertson and SparckJones 1976] takes the query-oriented view, and estimates the log ratio of relevance versus non-relevance given the document and the query. To instantiate this model, there is no need to directly represent the documents (e.g., using terms) – the only task is to represent queries in such a way that they will match well to those documents that the user will judge as relevant. The document-oriented view has been first proposed by Maron and Kuhn in [Maron and Kuhns 1960]. Here, documents are indexed by terms carefully chosen such that they will match well to the query to fulfill the user needs. Recently, the language modelling approach to information retrieval (e.g., [Ponte and Croft 1998]) builds upon the same document-oriented view. In the basic language models, a unigram model is estimated for each document and the likelihood of the document model with respect to the query is computed. Many variations and extensions have been proposed (e.g., [Hiemstra 2001; Lafferty and Zhai 2001; Zhai and Lafferty 2001]). The concept of relevance has been integrated into the language modelling approach to information retrieval by [Lavrenko and Croft 2001]. Several recent publications have further investigated the relationship between the RSJ models and the language modelling approach to information retrieval [Lafferty and Zhai 2003; Robertson 2005].

The two views rely on fixing one variable and optimizing the other variable, i.e., fixing the query and tuning the document or the other way around [Robertson 2003]. [Robertson 2003] has pointed out the drawback that neither view represents the problem of information retrieval completely. It seems intuitively desirable to treat both the query and the document as variables, and to optimize both. [Robertson et al. 1982] has illustrated this unified view to text retrieval using Fig. 1 (a). The authors identified three types of information that can be used for retrieval systems: 1) data describing the relations between other queries in the same class (or in other words similar queries) and this particular document, i.e. marginal information about the column; 2) data describing the relations between this particular query and other documents (similar documents), i.e. marginal information about the row; and 3) data describing the relations between other (similar) queries and other (similar) documents, i.e. the joint information about columns and rows.

The first literature proposing a unified model of information retrieval has been of a theoretical nature only [Robertson et al. 1982; Robertson 2003]. A first implementation of these ideas is found in [Bodoff 1999], where learning methods have been introduced into the vector space model to incorporate relevance information. Hereto, Multi-Dimensional Scaling (MDS) has been adopted to re-position the document vectors and the query vectors such that document representations are moved closer to queries when their relevance is observed, and away from queries when their non-relevance is observed. Bodoff and Robertson [2004] modeled the joint distribution of the observed documents, queries and relevances by assuming three underlying stochastic processes in the data generations: 1) the observed documents are generated by the true hidden document models, (2) the observed queries are generated by the true hidden query models, and (3) the relevances are generated by both the document and query models. A Maximum Likelihood (ML) method was applied to estimates the model parameters.

Going back to the collaborative filtering problem, Fig 1 (b) illustrates an analogy

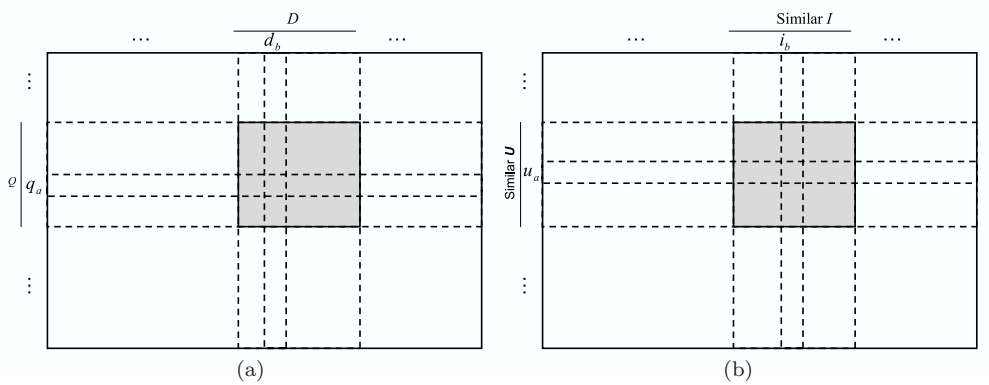



Fig. 1. Illustration of the joint data in text retrieval and collaborative filtering. (a) Query-document joint data in text retrieval (reproduced from [Robertson et al. 1982]). (b) User-item joint data in collaborative filtering.

between the concepts of users and items in CF and the concepts of documents and queries in information retrieval. Like Robertson et al. [1982] did for documents and queries in information retrieval, [Wang et al. 2006b] identified three types of information embedded in the user-item matrix (see also Section 3.3): 1) ratings of the same item by other users; 2) ratings of different items made by the same user, and, 3) ratings of other (but similar) items made by other (but similar) users. This paper explores deeper the usage of these three types of information in a unified probabilistic framework.

## 3. BACKGROUND

This section introduces briefly the user- and item-based approaches to collaborative filtering [Herlocker et al. 1999; Sarwar et al. 2001]. For $A$ users and $B$ items, the user profiles are represented in a $A \times B$ user-item matrix $\mathbf{X}$ (Fig. 2(a)). Each element $x_{a,b} = r$ indicates that user $a$ rated item $b$ by $r$, where $r \in \{1, ..., |R|\}$ and $|R|$ is the number of rating scales. if the item has been rated, and elements $x_{a,b} = \emptyset$ indicate that the rating is unknown.

The user-item matrix can be decomposed into row vectors,

$$\mathbf{X} = [\mathbf{u}_1, \ldots, \mathbf{u}_A]^T$$
$$\mathbf{u}_a = [x_{a,1}, \ldots, x_{a,B}]^T, \ a = 1, \ldots, A$$

corresponding to the $A$ user profiles $\mathbf{u}_a$. Each row vector represents the item ratings of a particular user. As discussed below, this decomposition leads to user-based collaborative filtering.

Alternatively, the matrix can be represented by its column vectors,

$$\mathbf{X} = [\mathbf{i}_1, ..., \mathbf{i}_B]$$
$$\mathbf{i}_b = [x_{1,b}, ..., x_{A,b}]^T, \ b = 1, ..., B$$

corresponding to the ratings by all $A$ users for a specific item $b$. As will be shown, this representation results in item-based recommendation algorithms.

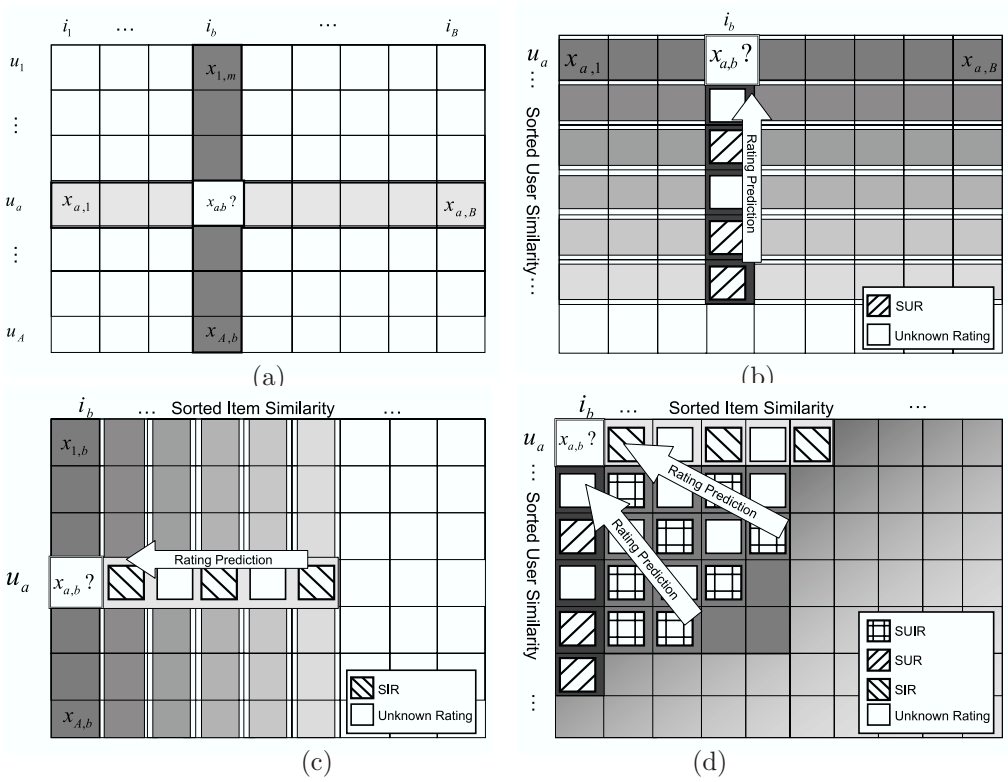



Fig. 2. (a) The user-item matrix (b) User-based approaches (c) Item-based approaches (d) Combining user-based and item-based approaches.

### 3.1 User-based Collaborative Filtering

User-based collaborative filtering predicts a test user's interest on a test item based on the ratings of *this* test item from other similar users. Ratings by more similar users contribute more to predicting the test item rating. The set of similar users can be identified by employing a threshold on the similarity measure or just selecting the top-$N$ most similar users. In the top-$N$ case, a set of top-$N$ users similar to test user $a$, $\mathcal{T}_{\mathbf{u}}(\mathbf{u}_a)$, can be generated by ranking users in order of their similarities:

$$\mathcal{T}_{\mathbf{u}}(\mathbf{u}_a) = \{\mathbf{u}_c | rank\ s_{\mathbf{u}}(\mathbf{u}_a, \mathbf{u}_c) < N\}, \tag{1}$$

where $s_{\mathbf{u}}(\mathbf{u}_a, \mathbf{u}_c)$ is the similarity between users $a$ and $c$. Cosine similarity and Pearson Correlation are popular similarity measures in collaborative filtering, see e.g. [Breese et al. 1998]. The similarity could also be learnt from training data [Cheung and Tian 2004; Jin et al. 2004]. Notice that the formulation in Eq. 1 assumes that $x_{c,b} = \emptyset \implies s_{\mathbf{u}}(\mathbf{u}_a, \mathbf{u}_c) = 0$, i.e., the users that did not rate the test item $b$ are not part of the top-$N$ similar users $\mathcal{T}_{\mathbf{u}}(\mathbf{u}_a)$.

Consequently, the predicted rating $\widehat{x}_{a,b}$ of test item $b$ by test user $a$ is computed

as

$$\widehat{x}_{a,b} = \overline{x}_a + \frac{\sum_{\mathbf{u}_c \in \mathcal{T}_\mathbf{u}(\mathbf{u}_a)} s_\mathbf{u}(\mathbf{u}_a, \mathbf{u}_c) \cdot (x_{c,b} - \overline{x}_c)}{\sum_{\mathbf{u}_c \in \mathcal{T}_\mathbf{u}(\mathbf{u}_a)} s_\mathbf{u}(\mathbf{u}_a, \mathbf{u}_c)}, \tag{2}$$

where $\overline{x}_a$ and $\overline{x}_c$ denote the average rating on all the rated items, made by users $a$ and $c$, respectively.

Existing methods differ in their treatment of unknown ratings from similar users (i.e. the value $x_{c,b} = \emptyset$). Missing ratings can be replaced by a 0 score, which lowers the prediction, or the average rating of that similar user could be used (mean imputation) [Breese et al. 1998; Herlocker et al. 1999]. Alternatively, [Xue et al. 2005] replaces missing ratings by an interpolation of the user's average rating and the average rating of his or her cluster (k-NN imputation).

Fig. 2(b) illustrates the user-based approach. In the figure, each user profile (row vector) is sorted and re-indexed by its similarity towards the test user's profile. Notice in Eq. 2 and in Fig. 2(b) user-based collaborative filtering takes only a small proportion of the user-item matrix into consideration for recommendation, i.e. only the *known test item ratings by similar users* are used. We refer to these ratings as the set of 'similar user ratings' (the blocks with upward diagonal pattern in Fig. 2(b)): $SUR_{a,b} = \{x_{c,b} | \mathbf{u}_c \in \mathcal{T}_\mathbf{u}(\mathbf{u}_a)\}$. For readability, we drop the subscript $a, b$ of $SUR_{a,b}$ in the remainder of the paper.

### 3.2 Item-based Collaborative Filtering

Item-based approaches such as [Deshpande and Karypis 2004; Linden et al. 2003; Sarwar et al. 2001] apply the same idea using the similarity between items instead of users. The unknown rating of a test item by a test user is predicted by averaging the ratings of other similar items rated by *this* test user. Ratings from more similar items are weighted stronger. Formally,

$$\widehat{x}_{a,b} = \frac{\sum_{\mathbf{i}_d \in \mathcal{T}_\mathbf{i}(\mathbf{i}_b)} s_\mathbf{i}(\mathbf{i}_b, \mathbf{i}_d) \cdot x_{a,d}}{\sum_{\mathbf{i}_d \in \mathcal{T}_\mathbf{i}(\mathbf{i}_b)} s_\mathbf{i}(\mathbf{i}_b, \mathbf{i}_d)}, \tag{3}$$

where item similarity $s_\mathbf{i}(\mathbf{i}_b, \mathbf{i}_d)$ can be approximated by the cosine measure or Pearson correlation [Linden et al. 2003; Sarwar et al. 2001]. To remove the difference in rating scale between users when computing the similarity, [Sarwar et al. 2001] has proposed to adjust the cosine similarity by subtracting the user's average rating from each co-rated pair beforehand. Like the top-$N$ similar users, a set of top-$N$ similar items towards test item $b$, denoted as $\mathcal{T}_\mathbf{i}(\mathbf{i}_b)$, can be generated according to:

$$\mathcal{T}_\mathbf{i}(\mathbf{i}_b) = \{\mathbf{i}_d | rank\ s_\mathbf{i}(\mathbf{i}_b, \mathbf{i}_d) < N\} \tag{4}$$

Fig. 2(c) illustrates the item-based approaches. Each item (column vector) is sorted and re-indexed according to its similarity towards the test item in the user-item matrix. Eq. 3 shows that only the *known similar item ratings by the test user* are taken into account for the prediction. We refer to the ratings used in the item-based

approach as the set of 'similar item ratings' (the blocks with downward diagonal pattern in Fig. 2(c)): $SIR_{a,b} = \{x_{a,d} | \mathbf{i}_d \in \mathcal{T}_{\mathbf{i}}(\mathbf{i}_b)\}$. Again, for simplicity, we drop the subscript $a, b$ of $SIR_{a,b}$ in the remainder of the paper.

### 3.3 Combining User-based and Item-based Approaches

When the ratings from these two sources are quite often not available, predictions are often made by averaging ratings from 'not-so-similar' users or items. Therefore, relying on $SUR$ or $SIR$ ratings only is undesirable.

In order to improve the accuracy of prediction, [Wang et al. 2006b] proposed to combine both user-based and item-based approaches. Additionally, we pointed out that the user-item matrix contains useful data beyond the previously used $SUR$ and $SIR$ ratings. As illustrated in Fig. 2 (d), the *similar item ratings made by similar users* may provide an extra source for prediction. They are obtained by sorting and re-indexing rows and columns according to their similarities towards the test user and the test item respectively. In the remainder, this part of the matrix is referred to as 'similar user item ratings' (the grid blocks in Fig. 2(d)): $SUIR_{a,b} = \{x_{c,d} | \mathbf{u}_c \in \mathcal{T}_{\mathbf{u}}(\mathbf{u}_a), \mathbf{i}_d \in \mathcal{T}_{\mathbf{i}}(\mathbf{i}_b), c \neq a, d \neq b\}$.

The subscript $a, b$ of $SUIR_{a,b}$ is dropped. Their similarity towards the target rating $x_{a,b}$, denoted as $s_{\mathbf{ui}}(x_{a,b}, x_{c,d})$, can be calculated as follows:

$$s_{\mathbf{ui}}(x_{a,b}, x_{c,d}) = \frac{1}{\sqrt{(1/s_{\mathbf{u}}(\mathbf{u}_a, \mathbf{u}_c))^2 + (1/s_{\mathbf{i}}(\mathbf{i}_b, \mathbf{i}_d))^2}} \tag{5}$$

where a Euclidean dis-similarity space is adopted such that the resulting combined similarity is lower than either of them. Now we are ready to combine these three types of ratings in a single collaborative filtering method. We treat each element of the user-item matrix as a separate predictor. Its confidence for the prediction is then estimated based upon its similarity towards the test rating. We then predict the test rating by averaging the individual predictions weighted by their confidence. Formally,

$$\widehat{x}_{a,b} = \sum_{x_{c,d}} p_{a,b}(x_{c,d}) W_{a,b}^{c,d}, \tag{6}$$

where

$$p_{a,b}(x_{c,d}) = x_{c,d} - (\overline{x}_c - \overline{x}_a) - (\overline{x}_d - \overline{x}_b) \tag{7}$$

can be treated as a normalization function when predicting rating $x_{a,b}$ from rating $x_{c,d}$. $\overline{x}_a$ and $\overline{x}_c$ are the average ratings by user $a$ and $c$, and $\overline{x}_b$ and $\overline{x}_d$ are the average ratings of item $b$ and $d$. For each test rating $x_{a,b}$, $W_{a,b}^{c,d}$ acts as a unified

weight matrix to combine the predictors from the three different sources:

$$W_{a,b}^{c,d} = \begin{cases} \frac{s_{\mathbf{u}}(\mathbf{u}_a,\mathbf{u}_c)}{\sum_{x_{c,d}\in SUR} s_{\mathbf{u}}(\mathbf{u}_a,\mathbf{u}_c)}\lambda(1-\delta) & x_{c,d} \in SUR \\ \frac{s_{\mathbf{i}}(\mathbf{i}_b,\mathbf{i}_d)}{\sum_{x_{c,d}\in SIR} s_{\mathbf{i}}(\mathbf{i}_b,\mathbf{i}_d)}(1-\lambda)(1-\delta) & x_{c,d} \in SIR \\ \frac{s_{\mathbf{ui}}(x_{a,b},x_{c,d})}{\sum_{x_{c,d}\in SUIR} s_{\mathbf{ui}}(x_{a,b},x_{c,d})}\delta & x_{c,d} \in SUIR \\ 0 & otherwise \end{cases}, \quad (8)$$

where $\lambda \in [0,1]$ and $\delta \in [0,1]$ control the importance of the different rating sources. This combination of ratings can be considered as two subsequent steps of linear interpolation. First, predictions from *SUR* ratings are interpolated with *SIR* ratings, controlled by $\lambda$. Next, the intermediate prediction is interpolated with predictions from the *SUIR* data, controlled by $\delta$. The second interpolation corresponds to smoothing the *SIR* and *SUR* predictions with *SUIR* ratings as a background model.

A bigger $\lambda$ emphasizes user correlations, while smaller $\lambda$ emphasizes item correlations. When $\lambda$ equals one, the algorithm corresponds to a user-based approach, while $\lambda$ equal to zero results in an item-based approach.

Tuning parameter $\delta$ controls the impact of smoothing from the background model (i.e. *SUIR*). When $\delta$ approaches zero, the fusion framework becomes the mere combination of user-based and item-based approaches without smoothing from the background model.

## 4. PROBABILISTIC RELEVANCE PREDICTION MODELS

This section re-formulates the collaborative filtering problem in a probabilistic framework. Motivated by the probabilistic relevance models proposed in text retrieval domain [Robertson and SparckJones 1976; Lafferty and Zhai 2003; Ponte and Croft 1998], we introduce the concept of "relevance" into collaborative filtering. By analogy with the relevance models in text retrieval, the collaborative filtering problem can be solved by answering the following basic question: what is the probability that *this* item is relevant to *this* user, given his or her profile?

To answer this question, firstly, let us define a sample space of relevance: $\Phi_R$ and let $R$ be a random variable over the relevance space $\Phi_R$. Unlike the commonly used binary relevance in text retrieval, in our case here, the relevance is multiple-valued. Thus, let us define that $\Phi_R$ has multiple values, which are observed from the rating data: $\Phi_R = \{1, ..., |R|\}$, where $|R|$ is the number of rating scales. From now on, each known element ($x_{a,b} \neq \emptyset$) in the user-item matrix is treated as an observation of this multiple-scaled relevance. Secondly, let $U$ be a discrete random variable over the sample space of *user id*'s: $\Phi_U = \{u_1, ..., u_A\}$ and let $I$ be a random variable over the sample space of *item id*'s: $\Phi_I = \{i_1, ..., i_B\}$, where $A$ is the number of users and $B$ the number of items in the collection. In other words, $U$ refers to the user identifiers and $I$ refers to the item identifiers.

We then denote $P$ as a probability function on the joint sample space $\Phi_U \times \Phi_I \times \Phi_R$. A prediction model thus can be built by estimating the conditional probability of the rating $P(R|U,I)$, given the specified user and item identifiers. The expectation of the unknown rating of a given item $I = i_b$ from a given user

$U = u_a$ can be formulated as follows:

$$\hat{x}_{a,b} = \sum_{r=1}^{|R|} rP(R = r|I = i_b, U = u_a) \tag{9}$$

For simplicity, $R = r$, $U = u_a$, and $I = i_b$ are denoted as $r$, $u_a$, and $i_b$, respectively.

Before proceeding, let us highlight how the current formulation using ratings differs from the probabilistic model that we have proposed before for log-based CF [Wang et al. 2006a]. In the log-based model, relevance variable $r$ is binary, observed as "the file is (not) downloaded" or the "web-site is (not) visited". In that case, $\Phi_R$ has binary values 'relevant' $r$ and 'non-relevant' $\overline{r}$. The resulting model is a *ranking model*, and does not attempt to predict the rating. Conversely, the model proposed in *this* paper is a rating prediction model, that is especially targeted to rating-based CF; i.e., the model predicts directly the number of stars that a user would assign to the test item.

### 4.1 Three Different Relevance Models

The way to estimate $P(r|i_b, u_a)$ plays an important role in our model. Three different models, namely *user-based relevance*, *item-based relevance* and *unified relevance* models can be derived if we apply the Bayes' rule differently:

$$P(r|i_b, u_a) = \begin{cases} \frac{P(u_a|r,i_b)P(r|i_b)}{P(u_a|i_b)} & \text{User-based Relevance} \\ \frac{P(i_b|r,u_a)P(r|u_a)}{P(i_b|u_a)} & \text{Item-based Relevance} \\ \frac{P(u_a,i_b|r)P(r)}{P(i_b,u_a)} & \text{Unified Relevance} \end{cases} \tag{10}$$

Each of the three derivations represents a different view of the problem. We shall see that the former two models (i.e. the user-based relevance model and the item-based relevance model) represent a partial view of the collaborative filtering problem while the third model unifies these partial views.

4.1.1 *User-based Relevance Model.* Applying the first factorization in Eq. 9, we derive:

$$\begin{aligned} \hat{x}_{a,b} &= \sum_{r=1}^{|R|} r \frac{P(u_a|r, i_b)P(r|i_b)}{P(u_a|i_b)} \\ &= \frac{\sum_{r=1}^{|R|} rP(u_a|r, i_b)P(r|i_b)}{P(u_a|i_b)} \\ &= \frac{\sum_{r=1}^{|R|} rP(u_a|r, i_b)P(r|i_b)}{\sum_{r=1}^{|R|} P(u_a|r, i_b)P(r|i_b)}, \end{aligned} \tag{11}$$

where the final prediction relies on two probabilities: 1) The conditional probability $P(u_a|r, i_b)$ builds up a user space model conditioned on the target item and the rating. It measures how probable a user may rate item $i_b$ as rating $r$ and thus it is the *preference model* of a user. 2) The probability $P(r|i_b)$ measures how likely the target item $i_b$ may be rated as rating $r$. It can be regarded as a rating *prior* or the *rating model*. Clearly, together the product of the two probabilities serves as a weight for each rating scale $r$. The denominator, a sum over the different rating

scales $\sum_{r=1}^{|R|} P(u_a|r,i_b)P(r|i_b)$, serves as a normalization factor. To simplify the notation, we choose $r_i$ to denote the joint event between $r$ and $i$ (i.e. rating for item $i$). Hence $P(u_a|r,i_b)$ is written as $P(u_a|r_{i_b})$.

Notice that, in the user preference model ($P(u_a|r,i_b)$) in Eq. 11, users are defined by their 'user identifiers' in the *user id* space. To be able to make a prediction for a new user, we need to build a feature representation and use it to relate the new user to the training users. For this, instead of putting users in the original *user id* space, we represent them by their ratings. So, $P(u_a|r_{i_b}) = P(\mathbf{u}_a|r_{i_b})$, where the vector $\mathbf{u}_a = [x_{a,1},\ldots,x_{a,B}]^T$ denotes the $B$ item ratings from user $a$. Unrated items can be filled with the average rating value, or taken as a constant value 0 instead. Substituting user identifiers by their rating vectors in Eq. 11 gives:

$$\hat{x}_{a,b} = \frac{\sum_{r=1}^{|R|} rP(\mathbf{u_a}|r_{i_b})P(r|i_b)}{\sum_{r=1}^{|R|} P(\mathbf{u_a}|r_{i_b})P(r|i_b)} \tag{12}$$

4.1.2 *Item-based Relevance Model.* We derive the following equation the same way, by factorizing $P(r|u_a,i_b)$ in Eq. 9 as $P(i_b|u_a,r)P(r|u_a)/P(i_b|u_a)$:

$$\hat{x}_{a,b} = \frac{\sum_{r=1}^{|R|} rP(i_b|r,u_a)P(r|u_a)}{\sum_{r=1}^{|R|} P(i_b|r,u_a)P(r|u_a)} \tag{13}$$

To simplify notation, let $r_u$ denote the joint event between r and u (i.e., the rating from user $u$), and $P(i_b|r,u_a)$ be written as $P(i_b|r_{u_a})$.

Following the same line of reasoning as for the user case above, an item can be represented using each user's rating as a feature, such that $P(i_b|r_{u_a}) = P(\mathbf{i}_b|r_{u_a})$ where the vector $\mathbf{i}_b = [x_{1,b},\ldots,x_{A,b}]^T$ denotes the $A$ user ratings for item $b$. Again, the missing ratings can be replaced by the average rating value or by a constant value 0. This gives

$$\hat{x}_{a,b} = \frac{\sum_{r=1}^{|R|} rP(\mathbf{i_b}|r_{u_a})P(r|u_a)}{\sum_{r=1}^{|R|} P(\mathbf{i_b}|r_{u_a})P(r|u_a)} \tag{14}$$

Probability $P(\mathbf{i_b}|r_{u_a})$ conditions the item space on the user's rating. It expresses how probable an item is rated as value $r$ by user $u_a$, and can be regarded as the *preference model* of an item. The probability $P(r|u_a)$ measures how likely the target user $u_a$ may provide a rating as value $r$. It can be treated as a rating *prior*, or, the target user's *rating model*. Obviously, the final weight for each rating scale is a product between these two models. The summation over different rating scales $\sum_{r=1}^{|R|} P(\mathbf{i_b}|r_{u_a})P(r|u_a)$ in the denominator serves as a normalization factor.

4.1.3 *Unified Relevance Model.* Let us now introduce the unified relevance model. We derive from Eq. 9:

$$\hat{x}_{a,b} = \frac{\sum_{r=1}^{|R|} rP(u_a,i_b|r)P(r)}{\sum_{r=1}^{|R|} P(u_a,i_b|r)P(r)} \tag{15}$$

Table I. Probabilities in the three different models.

| | Preference Model | Rating Model |
|---|---|---|
| User-based Relevance | $P(\mathbf{u_a}\|r_{i_b})$ | $P(r\|i_b)$ |
| Item-based Relevance | $P(\mathbf{i_b}\|r_{u_a})$ | $P(r\|u_a)$ |
| Unified Relevance | $P(\mathbf{u_a}, \mathbf{i_b}\|r)$ | $P(r)$ |

Analogous to the two other derivations, we use ratings as features to represent users and items, respectively: $P(u_a, i_b|r) = P(\mathbf{u}_a, \mathbf{i}_b|r)$.

$$\hat{x}_{a,b} = \frac{\sum_{r=1}^{|R|} rP(\mathbf{u_a}, \mathbf{i_b}|r)P(r)}{\sum_{r=1}^{|R|} P(\mathbf{u_a}, \mathbf{i_b}|r)P(r)} \tag{16}$$

The preference model now involves the user and item together.

Table I summarises the three different models derived from Eq. 9 (Eq. 12, 14 and 16). The weights to average the rating scales equal the product of a preference model and a rating model. Since the three models are derived from the same root, they are probabilistically equivalent, but the different factorizations lead to the different probability estimations, so statistically they are inequivalent.

## 4.2 Probability Estimation

The next problem is how to estimate the probabilities listed in Table I. Because the rating space is one-dimensional, the densities for the rating models can be estimated by simply counting the frequency of the co-occurrences. For the density estimations of the preference models however, the high dimensionality of the user feature space and the item feature space complicates matters. Here, we use the non-parametric Parzen-window method for density estimation (also known as kernel density estimation). This approach extrapolates the density from the sample data. The main advantage over a parametric approach is that we do not have to specify the model structure a priori, but may determine it from the data itself.

4.2.1 *Density Estimation for Rating Models.* Estimating the three rating models corresponds to counting the co-occurrence frequencies of the three joint events:

$$P(r|u_a) = \frac{\sum_{i_b} c(u_a, i_b, r)}{\sum_{i_b, r} c(u_a, i_b, r)} = \frac{|S_{r_{u_a}}|}{|S_{u_a}|} \tag{17a}$$

$$P(r|i_b) = \frac{\sum_{u_a} c(u_a, i_b, r)}{\sum_{u_a, r} c(u_a, i_b, r)} = \frac{|S_{r_{i_b}}|}{|S_{i_b}|} \tag{17b}$$

$$P(r) = \frac{\sum_{u_a, i_b} c(u_a, i_b, r)}{\sum_{u_a, i_b, r} c(u_a, i_b, r)} = \frac{|S_r|}{|S|}, \tag{17c}$$

where $c(u_a, i_b, r) \in \{0, 1\}$ denotes the co-occurrence function; $c(u_a, i_b, r)$ equals one if user $u_a$ rated item $i_b$ as $r$, zero otherwise. $S_{(\cdot)}$ denotes the set of observed samples where event $(\cdot)$ happened, $|S_{(\cdot)}|$ its cardinality. For example, $S_{r_{i_b}}$ denotes the set of observed samples with event $(R = r, I = i_b)$, so $|S_{r_{i_b}}|$ corresponds to the number of times that this event happened, $\sum_{u_a} c(u_a, i_b, r)$. $S$ denotes the entire set of observed samples and $|S|$ its size, equal to $\sum_{u_a, i_b, r} c(u_a, i_b, r)$.

Table II. Commonly used Parzen kernel functions.

| | |
|---|---|
| Rectangular | $\frac{1}{2}$ for $\|x\| < 1$, 0 otherwise |
| Triangular | $1 - \|x\|$for$\|x\| < 1$, 0 otherwise |
| Biweight | $\frac{15}{16}(1-x^2)^2$ for $\|x\| < 1$, 0 otherwise |
| Gaussian | $\frac{1}{\sqrt{2\pi}}e^{-x^2/2}$ |
| Bartlett - Epanechnikov | $\frac{3}{4}(1-x^2/5)/\sqrt{5}$ for $\|x\| < \sqrt{5}$, 0 otherwise |

4.2.2 *Density Estimation for Preference Models.* Given some observed samples of a population, the Parzen-window method extrapolates the data to the entire population by averaging from neighborhood observed samples. The neighborhood samples are selected and weighted according to some pre-defined Parzen kernel window functions. Usually, the Parzen kernel function is required to be non-negative and symmetric, and it should integrate to one. Table II lists the most commonly used Parzen kernels for univariate data. The multi-dimensional feature spaces in which users and items are represented require a multivariate Parzen kernel (denoted as $\mathbf{K}(\mathbf{y})$), that can be obtained using a product of univariate Parzen kernels:

$$\mathbf{K}(\mathbf{y}) = \prod_{q=1}^{Q} K_q(y_q), \tag{18}$$

where $\mathbf{y} = [y_1, ..., y_Q]$ is a Q-dimensional vector. $K_q$ is the univariate Parzen kernel function for the $q$th dimension. We assume all univariate kernels to be equal: $K_q = K$. The product of the univariate Parzen kernel assumes that the dimensions (features) are locally independent (where the locality is defined by the univariate Parzen kernel function $K$). Pluging in the bandwidth parameter, we have:

$$\frac{1}{h^Q}\mathbf{K}(\frac{\mathbf{y}}{h}) = \frac{1}{h^Q}\mathbf{K}(\frac{y_1}{h}, ..., \frac{y_Q}{h}) = \frac{1}{h^Q}\prod_{q=1}^{Q} K(\frac{y_q}{h}), \tag{19}$$

where to reduce the complexity of the model, we have assumed the bandwidth of each Parzen window to be equal in each dimension, defined as $h$.

First, take preference models $P(\mathbf{u}|r_{i_b})$ and $P(\mathbf{i}|r_{u_a})$ in the user-based and the item-based relevance derivations. Both depend on a single feature vector; either the user vector $\mathbf{u}$ or the item vector $\mathbf{i}$. Directly applying the Parzen-window method by using the univariate product Parzen kernel, we obtain the following equations for the density estimations ([Duda et al. 2001]):

$$P(\mathbf{u}|r_{i_b}) = \frac{1}{|S_{r_{i_b}}|}\sum_{\mathbf{u}' \in S_{r_{i_b}}} \frac{1}{{h_u}^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u}) \tag{20a}$$

$$P(\mathbf{i}|r_{u_a}) = \frac{1}{|S_{r_{u_a}}|}\sum_{\mathbf{i}' \in S_{r_{u_a}}} \frac{1}{{h_i}^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i}), \tag{20b}$$

where $h_i$ and $h_u$ are the bandwidth window parameters for the user vector and item vector, respectively.

Next, consider the preference model in the unified relevance case, $P(\mathbf{u_a}, \mathbf{i_b}|r)$. Probability estimation requires a density in the joint user-item feature space. We employ a product of two univariate kernel density estimators:

$$P(\mathbf{u}_a, \mathbf{i}_b|r) = \frac{1}{|S_r|} \sum_{(\mathbf{u}',\mathbf{i}')\in S_r} \frac{1}{{h_u}^B} \mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u}) \frac{1}{{h_i}^A} \mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i}), \tag{21}$$

where $S_r$ denotes the set of observed samples when event $(R = r)$ happens, while $|S_r|$ denotes its size, equal to $\sum_{u_a,i_b} c(u_a, i_b, r)$.

The kernel density estimation can be interpreted as a mixture of the component densities with an equal weight. Each mean of the component densities is located in the place where the neighborhood observation is available. The shape of the density is controlled by the Parzen kernel window function and its bandwidth parameter. Thus, the final prediction can be expressed by the summation over the Parzen kernels which are situated in the location of the neighborhood samples.

### 4.3 Rating Predictions

Consider now the problem of rating prediction in the user-based relevance model. Substituting the user-based rating model of Eq. 17a and the user preference model of Eq. 20a in the generic user-based relevance model of Eq. 12 gives:

$$\hat{x}_{a,b} = \frac{\sum_{r=1}^{|R|} r \frac{1}{|S_{r_{i_b}}|} \Big( \sum_{\mathbf{u}'\in S_{r_{i_b}}} \frac{1}{{h_u}^B} \mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u}) \Big) \frac{|S_{r_{i_b}}|}{|S_{i_b}|}}{\sum_{r=1}^{|R|} \frac{1}{|S_{r_{i_b}}|} \Big( \sum_{\mathbf{u}'\in S_{r_{i_b}}} \frac{1}{{h_u}^B} \mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u}) \Big) \frac{|S_{r_{i_b}}|}{|S_{i_b}|}} \tag{22}$$

Cancelling out the factors $|S_{r_{i_b}}|$ and $|S_{i_b}|$ simplifies Eq. 22 to:

$$\hat{x}_{a,b} = \frac{\sum_{r=1}^{|R|} r \Big( \sum_{\mathbf{u}'\in S_{r_{i_b}}} \mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u}) \Big)}{\sum_{r=1}^{|R|} \Big( \sum_{\mathbf{u}'\in S_{r_{i_b}}} \mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u}) \Big)} \tag{23}$$

Combining the outer summation over $r$ with the inner over $\mathbf{u}'$ completes our approach to perform rating prediction using the user-based relevance model:

$$\hat{x}_{a,b} = \frac{\sum_{\mathbf{u}'\in S_{i_b}} r_{\mathbf{u}',i_b} \mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})}{\sum_{\mathbf{u}'\in S_{i_b}} \mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})}, \tag{24}$$

where $\mathbf{u}' \in S_{i_b}$ denotes the (other) users who have rated item $i_b$, and, $r_{\mathbf{u}',i_b}$ denotes the rating of user $\mathbf{u}'$ for item $i_b$.

Rating prediction in the item-based relevance model is derived analogously:

$$\hat{x}_{a,b} = \frac{\sum_{\mathbf{i}'\in S_{u_a}} r_{\mathbf{i}',u_a} \mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})}{\sum_{\mathbf{i}'\in S_{u_a}} \mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})}, \tag{25}$$

where $\mathbf{i}' \in S_{u_a}$ denotes the other items rated by user $u_a$, and, $r_{\mathbf{i}',u_a}$ denotes the rating of user $\mathbf{i}'$ for item $u_a$.

Finally, rating prediction with the unified relevance model:

$$\hat{x}_{a,b} = \frac{\sum_{(\mathbf{u}',\mathbf{i}')\in S} r_{\mathbf{u}',\mathbf{i}'} \mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u}) \mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})}{\sum_{(\mathbf{u}',\mathbf{i}')\in S} \mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u}) \mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})}, \tag{26}$$

where $r_{\mathbf{u}',\mathbf{i}'}$ denotes the rating of user $\mathbf{u}'$ for item $\mathbf{i}'$. Again, $S$ denotes the entire sample set $S = S_1 \cup \cdots \cup S_{|R|}$.

The next step specifies the Parzen-window kernel function $K$ and its bandwidth parameter $h$. A common choice is to use the Gaussian density function as the univariate Parzen kernel function:

$$\frac{1}{h^Q}\mathbf{K}(\frac{\mathbf{y}}{h}) = \frac{1}{(\sqrt{2\pi}h)^Q} e^{-\frac{||\mathbf{y}||^2}{2h^2}} = \frac{1}{(\sqrt{2\pi}h)^Q} \prod_q e^{-\frac{||y_q||^2}{2h^2}} \tag{27}$$

Substituting the Gaussian Parzen-window in the above equations results in the following preference models:

$$\begin{aligned}
P(\mathbf{u_a}|r_{i_b}) &= \frac{1}{|S_{r_{i_b}}|(\sqrt{2\pi}h_u)^B} \sum_{\mathbf{u}' \in S_{r_{i_b}}} e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}} \\
P(\mathbf{i}|r_{u_a}) &= \frac{1}{|S_{r_{u_a}}|(\sqrt{2\pi}h_i)^A} \sum_{\mathbf{i}' \in S_{r_{u_a}}} e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}} \\
P(\mathbf{u}_a, \mathbf{i}_b|r) &= \frac{1}{|S_r|} \sum_{(\mathbf{u}',\mathbf{i}') \in S_r} \frac{1}{(\sqrt{2\pi}h_u)^B(\sqrt{2\pi}h_i)^A} e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}} e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}}
\end{aligned} \tag{28}$$

Using these three probability estimates gives the final three equations for rating prediction, corresponding to the three rating prediction models proposed in this paper:

User-based Relevance Model:

$$\hat{x}_{a,b} = \frac{\sum_{\mathbf{u}' \in S_{i_b}} r_{\mathbf{u}',i_b} \cdot e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}}}{\sum_{\mathbf{u}' \in S_{i_b}} e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}}} \tag{29a}$$

Item-based Relevance Model:

$$\hat{x}_{a,b} = \frac{\sum_{\mathbf{i}' \in S_{u_a}} r_{\mathbf{i}',u_a} \cdot e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}}}{\sum_{\mathbf{i}' \in S_{u_a}} e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}}} \tag{29b}$$

Unified Relevance Model:

$$\hat{x}_{a,b} = \frac{\sum_{(\mathbf{u}',\mathbf{i}') \in S} r_{\mathbf{u}',\mathbf{i}'} \cdot e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}} e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}}}{\sum_{(\mathbf{u}',\mathbf{i}') \in S} e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}} e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}}} \tag{29c}$$

### 4.4 Cross-validated EM algorithm

Previous studies have shown that the type of Parzen kernel function has usually only a marginal effect on the quality of density estimation; however, the choice of the bandwidth window parameter $h$ has a significant influence [Skurichina 1990]. In our case, the two bandwidth window parameters $h_u$ and $h_i$ need to be tuned to

the data. If their value is too small, the estimated density is a collection of sharp peaks positioned at the sample points, such that the density estimation still suffers from the data sparsity. If their value is however too large, the density estimate is over-smoothed, such that the underlying structure in the data is not preserved in the estimated density.

The optimal bandwidth parameters can be found by maximising the following cross-validated (leave the estimated $\mathbf{u}$ or $\mathbf{i}$ out) likelihood function [Duin 1976]:

$$\hat{h}_u, \hat{h}_i = \arg\max_{h_u, h_i} \prod_{(\mathbf{u},\mathbf{i}) \in S_r} \frac{1}{|S_r - 1|} \sum_{(\mathbf{u}',\mathbf{i}') \in S_r \wedge (\mathbf{u}' \neq \mathbf{u}) \wedge (\mathbf{i}' \neq \mathbf{i})} \frac{1}{h_u^B} \mathbf{K}\left(\frac{\mathbf{u} - \mathbf{u}'}{h_u}\right) \frac{1}{h_i^A} \mathbf{K}\left(\frac{\mathbf{i} - \mathbf{i}'}{h_i}\right) \tag{30}$$

This maximisation problem can be solved using the iterative Expectation maximisation (EM) algorithm [Dempster et al. 1977; Paclik et al. 2000]. For readability, we give the final expectation ($E$) and maximisation ($M$) steps here but leave the detailed derivation for Appendix A:

—$E$ step:

$$P^{(t)}(\mathbf{u}', \mathbf{i}' | \mathbf{u}, \mathbf{i}) = \frac{e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}} e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}}}{\sum_{\mathbf{u}' \neq \mathbf{u}, \mathbf{i}' \neq \mathbf{i}} e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}} e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}}} \tag{31a}$$

—$M$ step:

$$h_u^{(t+1)} = \sqrt{\frac{1}{B|S_r|} \sum_{\mathbf{u},\mathbf{i}} \sum_{\mathbf{u}' \neq \mathbf{u}, \mathbf{i}' \neq \mathbf{i}} P^{(t)}(\mathbf{u}', \mathbf{i}' | \mathbf{u}, \mathbf{i}) (||\mathbf{u} - \mathbf{u}'||^2)} \tag{31b}$$

$$h_i^{(t+1)} = \sqrt{\frac{1}{A|S_r|} \sum_{\mathbf{u},\mathbf{i}} \sum_{\mathbf{u}' \neq \mathbf{u}, \mathbf{i}' \neq \mathbf{i}} P^{(t)}(\mathbf{u}', \mathbf{i}' | \mathbf{u}, \mathbf{i}) (||\mathbf{i} - \mathbf{i}'||^2)} \tag{31c}$$

where $t$ equals the number of the iteration.

### 4.5 A Generalised Distance Measure

The univariate Gaussian Parzen kernel used in the previous measures the distance between users and item using Euclidean or $L_2$ distance ($||\mathbf{u} - \mathbf{u}'||^2$ and $||\mathbf{i} - \mathbf{i}'||^2$). However, many alternative distances could be considered. A previous study [Herlocker 2000] shows that the mean-squared difference is less effective for collaborative filtering than Pearson correlation and the cosine measure. We could of course adapt our framework using a different Parzen-window function, and try to set things up such that the density estimation is based on the cosine measure instead of Euclidean distance. Ideally, we would however like to *generalise* the Parzen-window density estimation from the specific distance measure of our choice.

We can achieve this goal using the mathematics that has become known as the *kernel trick* in the machine learning community [Schölkopf and Smola 2001]. Notice that the term kernel will refer to a *different* kernel function than the one in the Parzen-window density estimation; to avoid confusion, we use the term *projection kernel*. The kernel trick transforms any algorithm that solely depends on the dot product between two vectors, by replacing this dot product with the application of

the projection kernel. It has been proven (based on the Moore-Aronszajn-Theorem) that a positive definite projection kernel determines a unique function $\phi$ such that:

$$\mathcal{K}(\mathbf{y}, \mathbf{y}') =< \phi(\mathbf{y}), \phi(\mathbf{y}') >, \tag{32}$$

where $<,>$ denotes the dot product. This equation is often used without knowing the exact form of function $\phi$; it suffices to know that it exists and is defined uniquely. Schölkopf has shown that an even larger class of projection kernels (referred to as conditionally positive definite functions) satisfies Eq. 32 [Schölkopf 2000].

Now, basic linear algebra allows us to relate the Euclidean distance in projected space to the application of a projection kernel in user- or item-space:

$$\begin{aligned}
&||\phi(\mathbf{y}) - \phi(\mathbf{y}')||^2 \\
=&\sum_i (\phi(y_i) - \phi(y_i'))^2 \\
=&\sum_i \{\phi(y_i)^2 - 2\phi(y_i)\phi(y_i') + \phi(y_i')^2\} \\
=&\sum_i \phi(y_i)^2 + \sum_i \phi(y_i')^2 - 2\sum_i \phi(y_i)\phi(y_i') \\
=&< \phi(\mathbf{y}), \phi(\mathbf{y}) > + < \phi(\mathbf{y}'), \phi(\mathbf{y}') > -2 < \phi(\mathbf{y}), \phi(\mathbf{y}') > \\
=&\mathcal{K}(\mathbf{y}, \mathbf{y}) + \mathcal{K}(\mathbf{y}', \mathbf{y}') - 2\mathcal{K}(\mathbf{y}, \mathbf{y}')
\end{aligned} \tag{33}$$

Using a length-normalised projection kernel for which $\mathcal{K}(\mathbf{y}, \mathbf{y}) = 1$ gives

$$||\phi(\mathbf{y}) - \phi(\mathbf{y}')|| = \mathcal{K}(\mathbf{y}, \mathbf{y}) + \mathcal{K}(\mathbf{y}', \mathbf{y}') - 2\mathcal{K}(\mathbf{y}, \mathbf{y}') = 2 - 2\mathcal{K}(\mathbf{y}, \mathbf{y}') \tag{34}$$

So, computing a Euclidean distance in projected space is equivalent to using a positive definite projection kernel $\mathcal{K}$ to compute distances in the original space. This property allows us to perform Parzen-window density estimation with the Gaussian kernel in the projected space, without actually knowing the function $\phi$ (which is however defined uniquely by the choice of the projection kernel).

In the remainder, we use the length-normalised dot product as the projection kernel (also known as the cosine kernel, denoted as $Cos(\mathbf{y}, \mathbf{y}')$ [Liu et al. 2004]):

$$\mathcal{K}(\mathbf{y}, \mathbf{y}') = Cos(\mathbf{y}, \mathbf{y}') = \frac{< \mathbf{y}, \mathbf{y}' >}{\sqrt{< \mathbf{y}, \mathbf{y} >< \mathbf{y}', \mathbf{y}' >}} \tag{35}$$

Thus, we have:

$$||\phi(\mathbf{y}) - \phi(\mathbf{y}')||^2 = 2 - 2Cos(\mathbf{y}, \mathbf{y}') \tag{36}$$

In this specific case, function $\phi$ is actually known and corresponds to vector normalization:

$$\phi(\mathbf{y}) = \frac{\mathbf{y}}{\sqrt{< \mathbf{y}, \mathbf{y} >}} = \frac{\mathbf{y}}{\sqrt{\sum_i (y_i)^2}} \tag{37}$$

Eq. 36 demonstrates that the cosine *dissimilarity* measure is indeed equivalent to a Euclidean distance measure in the projected space. So, we can perform Parzen-window density estimation with a Gaussian window function in the projected space:

$$\mathbf{K}_h(\phi(\mathbf{y}), \phi(\mathbf{y}')) = \frac{J}{h} e^{-\frac{||\phi(\mathbf{y}) - \phi(\mathbf{y}')||^2}{2h^2}} = \frac{J}{h} e^{-\frac{1 - Cos(\mathbf{y}, \mathbf{y}')}{h^2}}, \tag{38}$$

Table III. Relationship between the choice of Parzen kernel, bandwidth parameters and the projection kernel and CF algorithms.

| Bandwidth Parameters | Parzen Kernel | Projection Kernel | CF Algorithm |
|---|---|---|---|
| $h_u \in \mathbb{R}$ | Gaussian | Cosine | User-based Relevance Model (Eq. 40a) |
| | | Null | Eq. 29a and PD ([Pennock et al. 2000]) |
| $h_i = \infty$ | Bartlett-Epanechnikov | Cosine | User-based method, VS ([Breese et al. 1998]) |
| | | Null | User-based, Ringo ([Shardanand and Maes 1995]) |
| $h_i \in \mathbb{R}$ | Gaussian | Cosine | Item-based Relevance Model (Eq. 40b) |
| | | Null | Eq. 29b |
| $h_u = \infty$ | Bartlett-Epanechnikov | Cosine | Item-based method, VS ([Sarwar et al. 2001]), |
| | | Null | Item-based Relevance Model, Euclidean Distance |
| $h_u \in \mathbb{R}$ | Gaussian | Cosine | Unified Relevance Model (Eq. 40c) |
| | | Null | Eq. 29c |
| $h_i \in \mathbb{R}$ | Bartlett-Epanechnikov | Cosine | ([Wang et al. 2006b]) and ([Hu and Lu 2006]) |
| | | Null | Unified Relevance Model, Euclidean Distance |

where $J$ is a normalization factor to obtain a Parzen window function, i.e., to satisfy

$$\int_{\mathbf{y}} \frac{J}{h} e^{-\frac{1-Cos(\mathbf{y},\mathbf{y}')}{h^2}} dy = 1 \tag{39}$$

It is easy to show that $J \in \mathbb{R}$ since $\frac{1}{h} e^{-\frac{1-Cos(\mathbf{y},\mathbf{y}')}{h^2}}$ is bounded in the $\mathbf{y}$ space.

By employing Eq. 38 instead of Eq. 27, we integrate the cosine similarity measure into our final rating prediction models:

User-based Relevance Model:

$$\hat{x}_{a,b} = \frac{\sum_{\mathbf{u}' \in S_{i_b}} r_{\mathbf{u}',i_b} e^{-\frac{1-Cos(\mathbf{u},\mathbf{u}')}{h_u^2}}}{\sum_{\mathbf{u}' \in S_{i_b}} e^{-\frac{1-Cos(\mathbf{u},\mathbf{u}')}{h_u^2}}} \tag{40a}$$

Item-based Relevance Model:

$$\hat{x}_{a,b} = \frac{\sum_{\mathbf{i}' \in S_{u_a}} r_{\mathbf{i}',u_a} e^{-\frac{1-Cos(\mathbf{i},\mathbf{i}')}{h_i^2}}}{\sum_{\mathbf{i}' \in S_{u_a}} e^{-\frac{1-Cos(\mathbf{i},\mathbf{i}')}{h_i^2}}} \tag{40b}$$

Unified Relevance Model:

$$\hat{x}_{a,b} = \frac{\sum_{(\mathbf{u}',\mathbf{i}') \in S} r_{\mathbf{u}',\mathbf{i}'} e^{-\frac{1-Cos(\mathbf{u},\mathbf{u}')}{h_u^2}} e^{-\frac{1-Cos(\mathbf{i},\mathbf{i}')}{h_i^2}}}{\sum_{(\mathbf{u}',\mathbf{i}') \in S} e^{-\frac{1-Cos(\mathbf{u},\mathbf{u}')}{h_u^2}} e^{-\frac{1-Cos(\mathbf{i},\mathbf{i}')}{h_i^2}}} \tag{40c}$$

Estimation of parameters $h_u$ and $h_i$ follows the same procedure as in the Euclidean distance case, see Appendix A.

### 4.6 Discussion

**The General Framework** The proposed combination of Parzen-window kernel density estimation with the relevance models provides a general framework for col-

laborative filtering. The three rating prediction models listed in Eq. 40a, 40b and 40c, show how the final predictions are expressed by summations over rating influences from the neighborhood samples (from user neighbors Eq. 40a, item neighbors Eq. 40b, or both the user and item neighbors Eq. 40c). Three factors determine the influence of the neighborhood samples on the prediction: the type of Parzen kernel, its bandwidth parameters, and, the distance of the neighborhood samples from the sample to be predicted. The Parzen kernel smoothes the prediction, while the projection kernel allows us to select the right distance measure. Different choices for the bandwidth parameters, the Parzen-window kernel function or the projection kernel lead to different approaches to collaborative filtering. For instance, it is easy to see that using the Bartlett-Epanechnikov kernel (given in Table II) with $h_i$ equal to one and $h_u$ to $\infty$ simplifies the unified relevance prediction formula in Eq. 26 to the item-based cosine similarity method (using Eq. 36):

$$\hat{x}_{a,b} = \frac{\sum_{\mathbf{i}' \in S_{u_a}} r_{\mathbf{i}',u_a} Cos(\mathbf{i}, \mathbf{i}')}{\sum_{\mathbf{i}' \in S_{u_a}} Cos(\mathbf{i}, \mathbf{i}')} \tag{41}$$

Table III summarises how the general framework can be specialised to various previously known approaches.

**The User-based and Item-Based Views** The user-based relevance model and item-based relevance model represent two different views for the problem. In the user-based relevance model, we fix the target item $i_b$. Conditioned on it, we build up a user representation $P(\mathbf{u_a}|i_b, r)$ (See Fig. 3 (a)). This is analogous to the document-oriented approaches in text retrieval [Maron and Kuhns 1960; Lafferty and Zhai 2003], where queries represented by the terms are conditioned on the fixed target document model $P(\mathbf{Q}|d_b, r)$.[1] Conversely, in the item-based relevance model, we fix the target user $u_a$ and conditioned on it, we build up an item representation $P(\mathbf{i_b}|u_a, r)$ (see Fig. 3 (b)). This is analogous to the query-oriented approach in text retrieval [Robertson and SparckJones 1976], where documents $\mathbf{D}$ are represented by the terms and these representations are conditioned on the fixed query terms $P(\mathbf{D}|q_a, r)$.[2]

**The Unified View** Unlike the above two models, the unified relevance model in Eq. 40c however provides a rather completed and unified view of the problem. In this model, we do not fix the two variables: user and item. Instead, we construct a unified model that relies on both the user representation and the item representation $P(\mathbf{u_a}, \mathbf{i_b}|r)$ (see Fig. 3 (c)). The model is solved by applying the kernel density estimation and it provides a practical solution for the unification advocated in [Robertson et al. 1982]. It can also be treated as a generalised version of the similarity fusion approach to CF [Wang et al. 2006b].

The model intuitively provides a unified probabilistic framework to fuse user-based and item-based approaches. In addition, the ratings from similar users for similar items (the SUIR ratings) are employed to smooth the predictions. We highlight the relationship to the similarity fusion method of [Wang et al. 2006b] by

[1] In the language modelling literature, the document-oriented approach is also referred to as a *query-generation* model.

[2] The query-oriented approach is also known as a *document-generation* model for information retrieval.

dividing the entire set of observations $S$ into three sets: the ratings from test user $S_{\mathbf{u}}$, the ratings for test item $S_{\mathbf{i}}$, and the remaining ratings $S_{\bar{\mathbf{u}},\bar{\imath}}$. Eq. 40c gives

$$\begin{aligned}\hat{x}_{a,b} &= \frac{1}{H}\Big(\sum_{(\mathbf{u}',\mathbf{i}')\in S} r_{\mathbf{u}',\mathbf{i}'} e^{-\frac{1-Cos(\mathbf{u},\mathbf{u}')}{h_u^2}} e^{-\frac{1-Cos(\mathbf{i},\mathbf{i}')}{h_i^2}}\Big) \\ &= \frac{1}{H}\Big(\sum_{(\mathbf{u}',\mathbf{i})\in S_{\mathbf{i}}} r_{\mathbf{u}',\mathbf{i}} E + \sum_{(\mathbf{u},\mathbf{i}')\in S_{\mathbf{u}}} r_{\mathbf{u},\mathbf{i}'} F + \sum_{(\mathbf{u}',\mathbf{i}')\in S_{\bar{\mathbf{u}},\bar{\imath}}} r_{\mathbf{u}',\mathbf{i}'} G\Big),\end{aligned} \tag{42}$$

where $H$ is a normalization factor equal to

$$\sum_{(\mathbf{u}',\mathbf{i}')\in S} e^{-\frac{1-Cos(\mathbf{u},\mathbf{u}')}{h_u^2}} e^{-\frac{1-Cos(\mathbf{i},\mathbf{i}')}{h_i^2}} \tag{43}$$

The three types of ratings $r_{\mathbf{u}',\mathbf{i}}$, $r_{\mathbf{u},\mathbf{i}'}$ and $r_{\mathbf{u}',\mathbf{i}'}$ that contribute to the prediction are precisely the similar user ratings (SUR), the similar item ratings (SIR) and the similar user towards similar item ratings (SUIR). E, F and G determine three weights for averaging these ratings:

$$\begin{aligned}E &= e^{-\frac{1-Cos(\mathbf{u},\mathbf{u}')}{h_u^2}} \\ F &= e^{-\frac{1-Cos(\mathbf{i},\mathbf{i}')}{h_i^2}} \\ G &= e^{-\frac{1-Cos(\mathbf{u},\mathbf{u}')}{h_u^2}} e^{-\frac{1-Cos(\mathbf{i},\mathbf{i}')}{h_i^2}}\end{aligned} \tag{44}$$

The bandwidth parameters $h_u$ and $h_i$ control the width of the kernel function. A small bandwidth value leads to spiky estimates while larger bandwidth values over-smooth the observations with data from far away samples. The bandwidth parameters also balance the contributions from the user side and the item side. A small $h_u$ emphasizes user correlations, and a small $h_i$ emphasizes the item correlations (see also the experiments corresponding to Fig. 9 and Fig. 10). When $h_u$ approaches $\infty$, the unified relevance model corresponds to item-based collaborative filtering, while an $h_i$ of $\infty$ results in user-based recommendation.

### 4.7 Computational Complexity

This section discusses the scalability of our collaborative filtering framework. The computational complexity of the framework consists of the amount of time needed for building the model (i.e. the EM estimation of the two bandwidth parameters and the probability estimations), and that of making online recommendations for a new user from the model. The EM algorithm is only needed during the model building phase. Thus there are no iterative steps required in the online recommendation phase. Furthermore, we propose an efficient method to calculate the kernel-based similarities, largely reducing the computational complexity.

4.7.1 *Offline Computation.* In the model building phase, Eq. 29c simplifies the probability estimations to the calculations of the kernel-based similarities. That is, for any user pair and item pair, we need to calculate their kernel similarities $e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}}$ and $e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}}$, respectively. To make the calculation efficient, we decompose them into the distance measure part ($||\mathbf{u}-\mathbf{u}'||$ or $||\mathbf{i}-\mathbf{i}'||$) and the kernel

smoothing part ($e^{-\frac{\bullet}{2h_u{}^2}}$ or $e^{-\frac{\bullet}{2h_i{}^2}}$). Thus, model building consists of two steps: first computing two distance (dis-similarity) matrices and then estimating the two bandwidth parameters.

For each element in the matrix, we require either B arithmetic operations for the user-to-user distance or A arithmetic operations for the item-to-item distance. In total the upper bound of the computational complexity is $O(A^2B + B^2A)$, which is roughly equal to a sum of the complexity of a user-based method [Herlocker et al. 1999] and that of an item-based method [Sarwar et al. 2001]. Since the data is extremely sparse, with a proper data structure, the computation can be largely reduced. This paper proposes to use two inverted files, respectively indexing users and items about their ratings. When we calculate the distances, for instance, for a given user, we do not need to go through *all* other users about their agreement on *all* items. Instead, we first from the user indexing file get the set of items that he or she has rated, and then go through these items, accessing a set of users who have rated any of these items (from the item index file). By doing this, not only do we exclude the users who do not share any commonly-rated item from the computations, we also restrict the operations to those items that the two users both rated. Thus the overall computation is much faster than the original user-based or item-based methods, only requiring a linear time complexity that approximately equals $O((A+B)mn)$, where $m$ is the average number of user ratings per item and $n$ is the average number of item ratings per user. In addition, it is unnecessary to store all the non-zero elements in the distance matrices, as we shall see in our experiment (Fig. 8) that keeping only the top-$N$ nearest user neighbors and the top-$N$ nearest item neighbors improves prediction accuracy, where $N$ is typically in the range of (30...70).

Once we have the user (item) distance matrix that stores the top-$N$ nearest neighbors, the EM estimation of $h_i$ and $h_u$ becomes a relatively simple task because it essentially averages the user or item distance from the distance matrices in an iterative manner (see Eq. 31). For the sake of time efficiency, the E step and M step can be computed together, where the computational complexity in each iteration is given by $O(ABN^2)$ because we need to average all possible user-item pair ($A \times B$ operations) and for each pair, we need to access $N \times N$ neighbors. In practice, the complexity of the EM algorithm can be further reduced by sub-sampling both training users and training items; our additional experiments (not reported) verified that a small amount of training user-item pairs is sufficient to get the stable and accurate estimations of $h_u$ and $h_i$. We shall see in our experiment (Fig. 4) that the EM algorithm converges fast, and a small number of iterations (3-5) suffices to get relatively stable estimations in the tested data sets.

4.7.2 *Online Computation.* The online recommendation can be computed very efficiently if we again utilize both the user and item indexing files. For a new user, the computational complexity of his or her kernel similarity towards other users is approximately given as $O(mn)$ because on average we access his or her $m$ rated items and for each of these items access $n$ users who rate it. The final prediction corresponds to a weighted average from the ratings of the top-$N$ nearest items and the top-$N$ nearest users, resulting in a computational complexity of $O(N^2)$, which is independent of the number of users and items.

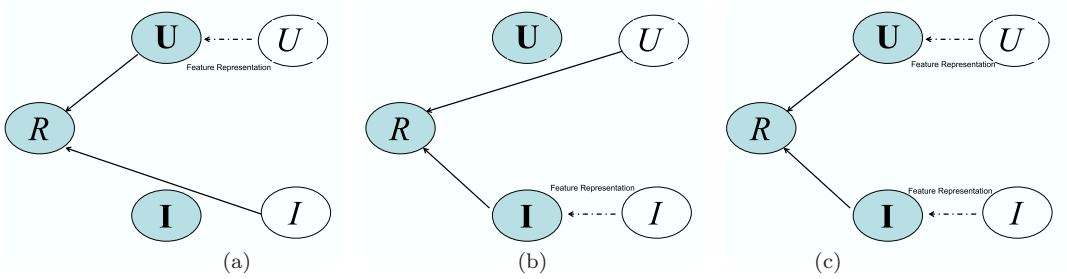



Fig. 3. Illustration of the three different models. (a) User-based Relevance Model. (b) Item-based Relevance Model. (c) Unified Relevance Model.

Table IV. Characteristics of the test data sets.

| | MovieLens 1 | MovieLens 2 | EachMovies 1 | EachMovies 2 |
|---|---|---|---|---|
| Num. of Users | 943 | 500 | 2,000 | 10,000 |
| Num. of Items | 1682 | 1000 | 1,648 | 1,648 |
| Avg. Num. of Rated Item Per User | 106.0 | 87.7 | 90.0 | 96.3 |
| Avg. Num. of User Rating Per Item | 59.5 | 43.9 | 114.2 | 611.0 |
| Sparsity | 6.30% | 8.77% | 5.7% | 6.11% |
| Rating Scales | 5(1-5) | 5(1-5) | 6(1-6) | 6(1-6)) |

## 5. EXPERIMENTS

### 5.1 Data Sets

We experimented with two movie rating data sets: the MovieLens [DataSet a] and the EachMovie [DataSet b] data sets.

The MovieLens data set was collected by the GroupLens group through the MovieLens web site during the period from September 1997 through April 1998. It contains ratings by 943 users for 1682 movies (items). Each user has rated at least 20 movies. The rating scale takes values from 1 ( the lowest rating) to 5 (the highest rating). In addition, to compare with other approaches we also adopt a widely-used subset [Si and Jin 2003], which contains 500 users and 1000 movies (items), where each user has rated at least 40 items.

The EachMovie data set was collected by the Digital Equipment Research Center during the period from 1995 to 1997. The rating scale was originally indicated as the values from 0 (no star), 0.2 (one star) and up to 1 (five stars). For consistency with the MovieLens data set, we transformed the rating scales to the values $1--6$, with 1 being the lowest rating (i.e., no star) and 6 being the highest one (i.e., five stars). To compare with other approaches, we adopt the two subsets described in [Si and Jin 2003] and [Xue et al. 2005], which respectively contain 2,000 users and 10,000 users. In both cases, each user has rated as least 40 items. The basic characteristics of these two data sets with the different size are summarised in Table IV. We mainly use the MovieLens 1 data set to conduct empirical analyses on our models while using the other sets to conduct the performance tests.

### 5.2 Evaluation Protocols

For testing, we assigned the users in the data set at random to a training user and a test user set. Users in the training set are used as the basis for making predictions, while our test users are considered the ground truth for measuring prediction accuracy. Each test user's ratings have been split into a set of *observed items* and one of *held-out items*. The ratings of the observed items are input for predicting the ratings of the held-out items (the test items). To improve our measurements, each of the experimental setups has been repeated 20 times with different random seeds.

For consistency with experiments reported in the literature, e.g., [Jin et al. 2004; Sarwar et al. 2001; Xue et al. 2005]), we report our results using the mean absolute error (MAE) evaluation metric. MAE corresponds to the average absolute deviation of predictions to the ground truth data, for all test item ratings and test users:

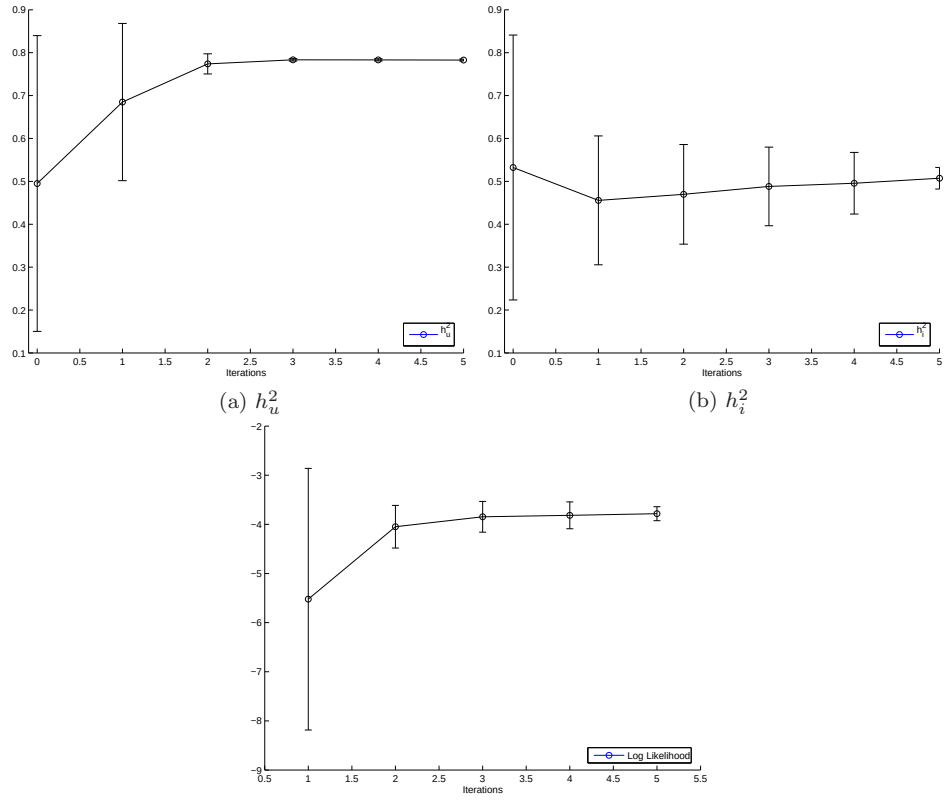
$$MAE = \frac{\sum_{a,b} |x_{a,b} - \widehat{x}_{a,b}|}{L}, \tag{45}$$

where $L$ denotes the number of tested ratings. A smaller value indicates a better performance.

### 5.3 Results

5.3.1 *Parameter Estimation.* This section conducts the experiments on the EM estimation for the parameters $h_u$ and $h_i$. We first test the convergence behavior of the cross-validated EM method using MovieLens set 1. To reduce redundancy, we only show the estimation results when we randomly select 400 users as the training data. Notice that the estimation over other number of training users behaves consistently. Fig. 4 shows that the EM algorithm converges in few iterations (about 3) to the optimal bandwidth parameters ($h_u^2 = 0.79$ (Fig. 4(a)) and $h_i^2 = 0.49$ (Fig. 4(b))) with respect to the log likelihood object function (Fig. 4 (c)). Repeating this experiment with different random initial values of $h_u$ and $h_i$ (have a relatively large standard deviation in the figures), the EM algorithm converges to (almost) the same optimal values (have a relatively small standard deviation in the figures). Observe that the obtained bandwidth parameter $h_u$ is relatively larger than the parameter $h_i$. To explain this result, we investigate the influence of the distribution of neighboring vectors on the parameter estimation. In our models, both the user distance and item distance are measured by the cosine distance, i.e. $1 - Cos(\mathbf{u}, \mathbf{u}')$ and $1 - Cos(\mathbf{u}, \mathbf{u}')$ (Eq. 40a–40c). Fig. 5 plots the distributions of the top-50 nearest users and items as a histogram of 10 bins in the distance range of $[0, 1]$. It shows that, on average, the user distances are larger than the item distances. So, estimating the user density needs a bigger bandwidth parameter to smooth from the neighborhood than that of the item density.

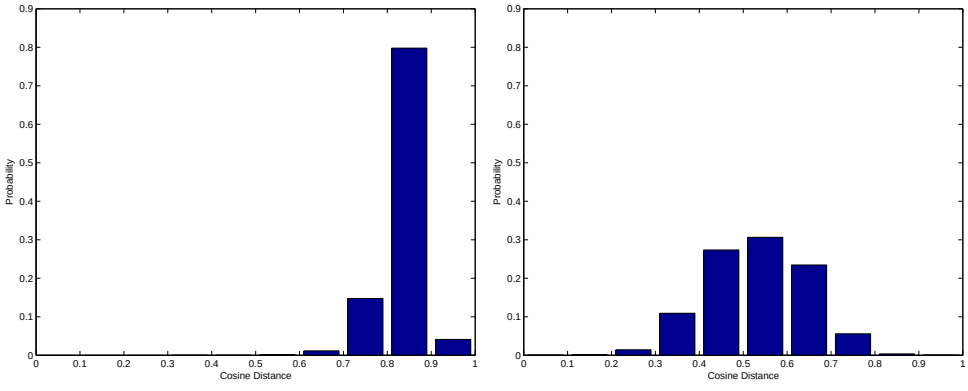
To show the sensitivity of the two bandwidth parameters regarding to the recommendation performance, we plot the value of the bandwidth parameter (either $h_u$ or $h_i$) against the MAE measurement in Fig. 6. It shows that the two bandwidth parameters are relatively stable in a wide range regarding to the MAE performance. Also, comparing the optimal bandwidth parameters with the ones shown in Fig. 4 we can see that, although the EM algorithm uses a different objective function




(a) $h_u^2$ (b) $h_i^2$

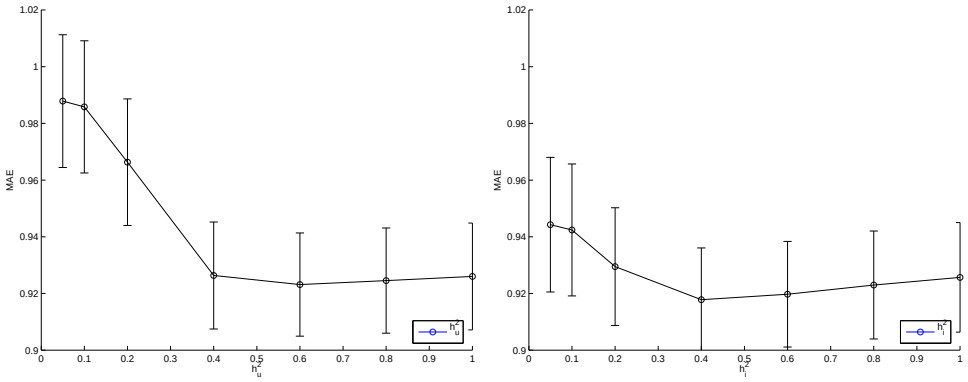
(c) Log Likelihood Objective Function

Fig. 4. Convergence behavior of the cross-validated EM algorithm: Using the cosine projection kernel (400 training users in the MovieLens 1 data set).



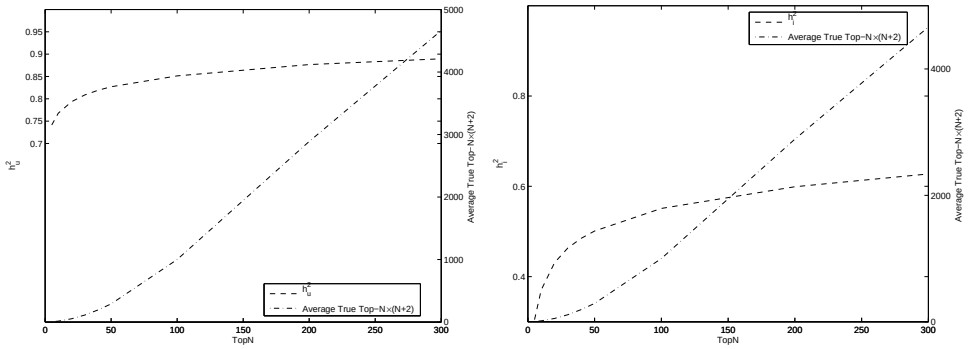

(a)Top-50 User (b)Top-50 Item

Fig. 5. Distribution of cosine distance in the MovieLens data set (400 training users in the MovieLens 1 data set).




(a)$h_u^2$ (b)$h_i^2$

Fig. 6. The sensitivity of the two parameters regarding to the MAE measurement (the remaining 543 test users in the MovieLens 1 data set).




(a) $h_u^2$ vs. top-N (b) $h_i^2$ vs. top-N

Fig. 7. The impact of the neighbor size on the parameter estimation. (400 training users in the MovieLens 1 data set).

(log likelihood), it does give a reasonably good estimation of the two bandwidth parameters in terms of the MAE.

In practice, recommendation systems make a trade-off between prediction accuracy and run-time system efficiency by pre-selecting the top-$N$ nearest user neighbors (SUR) and the top-$N$ nearest item neighbors (SIR). These form a rating pool, extended with the top-$N \times N$ nearest similar user to similar item neighbors (SUIR). In total, we would have $N \times (N+2)$ neighbors. However, only the neighbors whose ratings are available in the pool contribute to the predictions. Clearly, the parameter $N$ affects the parameter estimation and therefore also the performance of our fusion methods. Fig. 7 plots the estimated parameter values of $h_u$ and $h_i$ under different neighbor sizes, where the $x$ axis represents the size of the pre-selected top-$N$ and the left $y$ axis corresponds to the estimated value of the parameter ($h_u$ in Fig. 7(a) and $h_i$ in Fig. 7(b)). The right $y$ axis shows the 'true' number of SUR, SIR and SUIR within the pool that have to be retrieved to obtain non-empty ratings for estimation. The graph shows that for low values of $N$, both parameters $h_u$

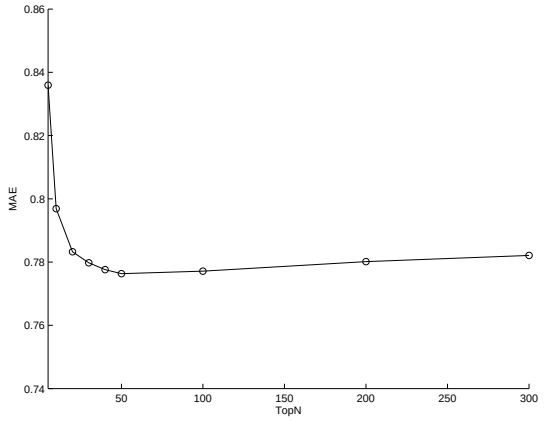
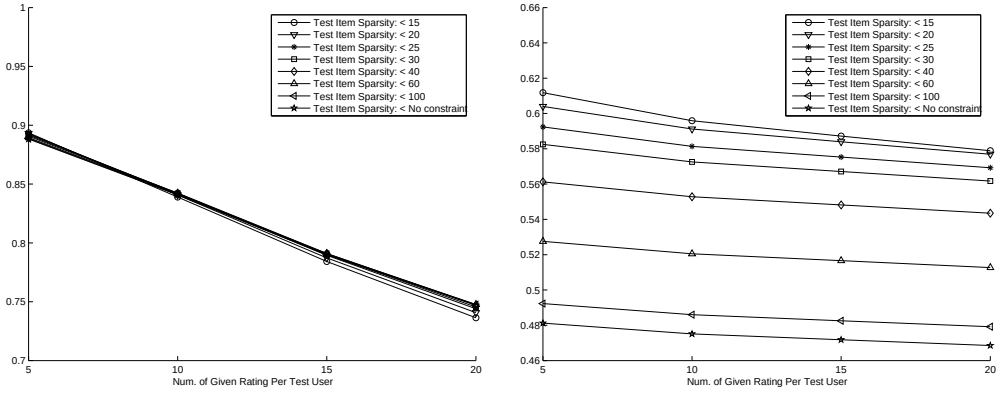



Fig. 8. The MAE performance under different neighbor size (the remaining 543 test users in the MovieLens 1 data set).

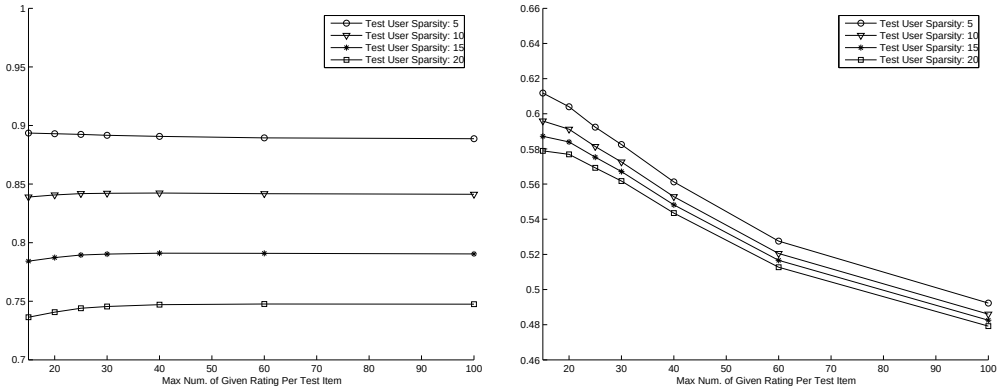



(a) $h_u^2$ (b) $h_i^2$

Fig. 9. The optimal bandwidth parameters with different user sparsity.

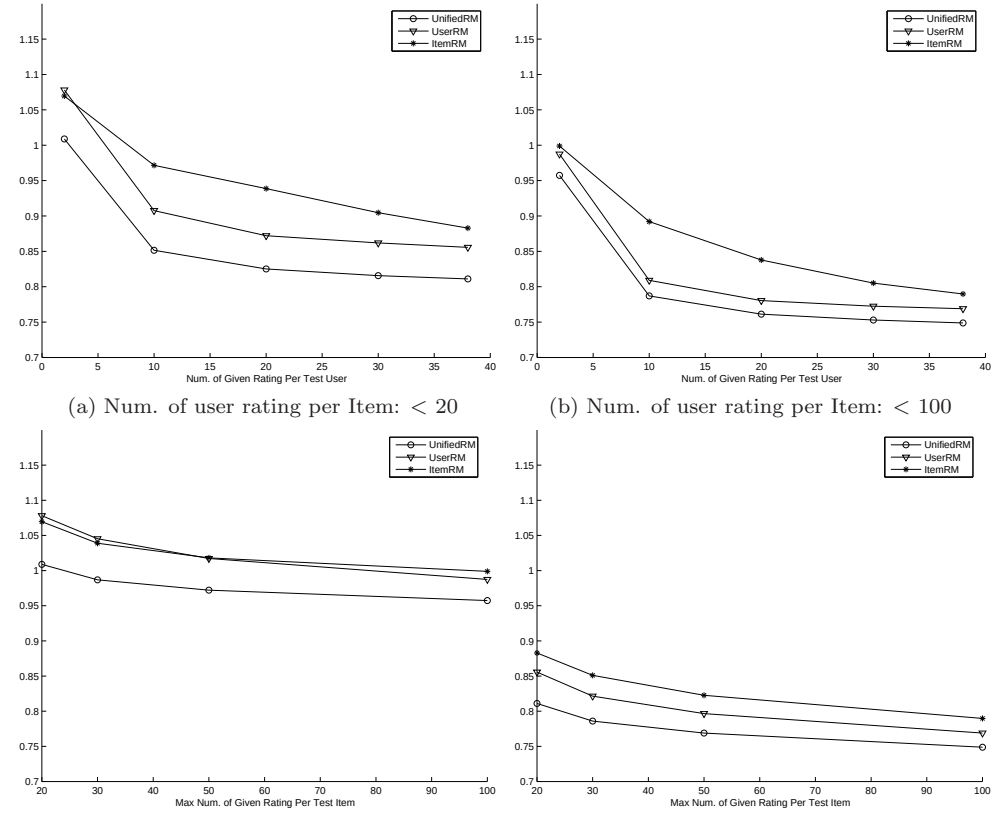
and $h_i$ increase fast. This shows that the new ratings introduced by increasing the top-$N$ contribute to improve the prediction accuracy, such that the corresponding Parzen-window should cover the newly introduced ratings (so, the bandwidth parameters increase). Gradually however, this increase diminishes, because a large top-$N$ introduces more and more noisy ratings. Consequently, the parameter values converge and the Parzen-window excludes the distant ratings from the estimation process. Fig. 8 displays the MAE of the unified relevance model under different neighbor sizes, to illustrate the effect of the estimated parameters on prediction accuracy. The optimal result corresponds to $N \simeq 50$. The error increases only slowly with larger values of $N$, due to the fact that the Parzen-window reduces the effect of the noisy (distant) neighbors when they are introduced. We select 50 as the optimal choice of $N$ for the subsequent experiments.

5.3.2 *Sparsity.* This section investigates the effect of data sparsity on the performance of our collaborative filtering methods in more detail. For this, we set up




(a) $h_u^2$ (b) $h_i^2$

Fig. 10. The optimal bandwidth parameters under different item sparsity.

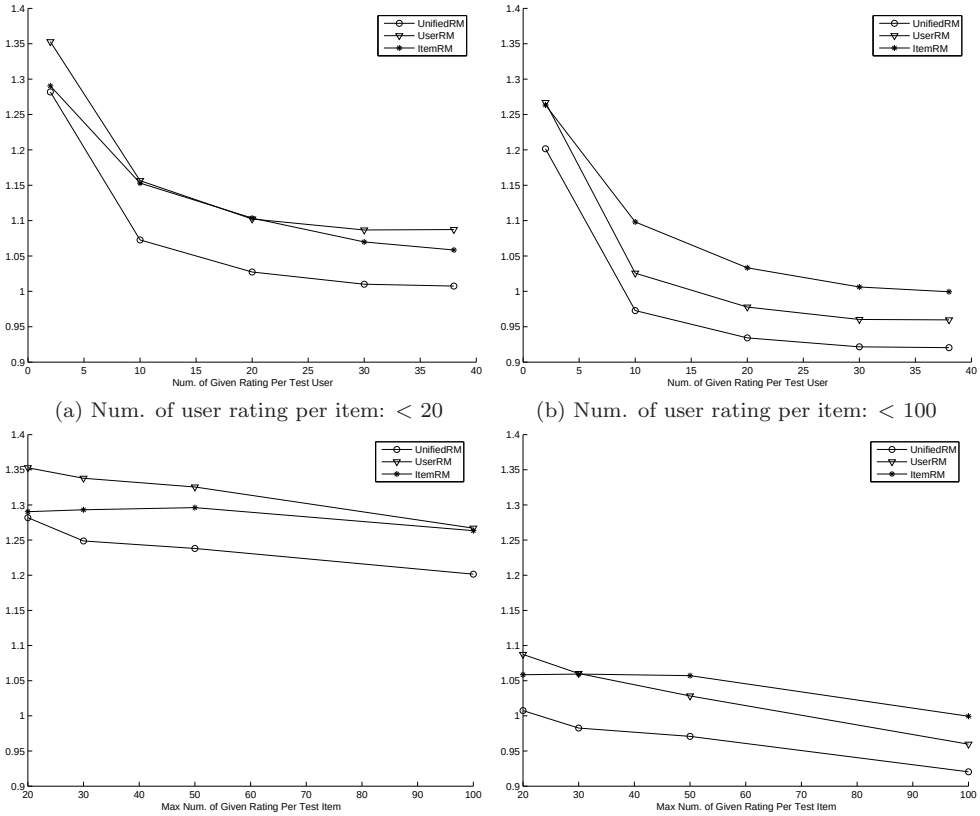



(a) Num. of user rating per Item: $< 20$ (b) Num. of user rating per Item: $< 100$

(c) Num. of item rating per user: 2 (d) Num. of item rating per user: 38

Fig. 11. Performance of the three derived models under different sparsity in the MovieLens 1 data set.

the following configurations: 1) *Test User Sparsity*: vary the number of items rated by test users in the observed set, e.g. 5, 10, or 20 ratings per user. 2) *Test Item*




(a) Num. of user rating per item: $< 20$

(b) Num. of user rating per item: $< 100$

(c) Num. of item rating per User: 2

(d) Num. of item rating per User: 38

Fig. 12. Performance of the three derived models under different sparsity in the EachMovie 1 data set.

*Sparsity*: vary the number of users who have rated test items in the held-out set, e.g. less than 15, 20, 25 (denoted as '$<$ 15', '$<$ 20', or '$<$ 25'), or, unconstrained (denoted as 'No constraint'). Notice that the configurations of the user sparsity and the item sparsity are not completely symmetrical in order to reflect the practical situation.

The first experiment investigates the effect of data sparsity on parameter estimation using the EM algorithm. We use the different sparsity configurations: user sparsity: number of given ratings per user (5, 10, 15, 20) and item sparsity: maximum number of user rating per item ($<$15, $<$20, $<$25, $<$30, $<$40, $<$60, $<$100, *No constraint*). For each configuration, we select 400 users in the MovieLens data set to run the EM algorithm to obtain the optimal parameters $h_i$ and $h_u$.

Fig. 9 (a) and (b) show, respectively, the optimal values of $h_u^2$ (y axis in Fig. 9 (a)) and $h_i^2$ (y axis in Fig. 9 (b)) under different user sparsity conditions (x axis). The figures demonstrate that when the user sparsity decreases (and therefore the number of given ratings per user increases), the optimal user bandwidth parameter $h_u$ becomes smaller while the optimal item bandwidth parameter $h_i$ stays relatively constant. This is due to the fact that, for a given test user, when the number of

Table V. Comparison among the three derived models on the MovieLens 1 data set. The MAE and P-value of the 10-fold Wilcoxon signed-rank test are reported (the minimum P-value for 10-fold is 0.002).

| **User Sparsity** | **5** | **10** | **20** | **30** | **38** |
|---|---|---|---|---|---|
| **Unified RM** | **1.009** | **0.851** | **0.825** | **0.816** | **0.811** |
| **User-based RM** | 1.078 | 0.908 | 0.872 | 0.862 | 0.856 |
| **Item-based RM** | 1.070 | 0.972 | 0.939 | 0.905 | 0.883 |
| - | **P-value** | **P-value** | **P-value** | **P-value** | **P-value** |
| **Unified - UserRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |
| **Unified - ItemRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |

(a) Num. of user rating per item: <20

| **User Sparsity** | **5** | **10** | **20** | **30** | **38** |
|---|---|---|---|---|---|
| **Unified RM** | **0.987** | **0.826** | **0.799** | **0.791** | **0.786** |
| **User-based RM** | 1.045 | 0.869 | 0.836 | 0.826 | 0.821 |
| **Item-based RM** | 1.039 | 0.947 | 0.905 | 0.872 | 0.851 |
| - | **P-value** | **P-value** | **P-value** | **P-value** | **P-value** |
| **Unified - UserRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |
| **Unified - ItemRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |

(b) Num. of user rating per item: <30

| **User Sparsity** | **5** | **10** | **20** | **30** | **38** |
|---|---|---|---|---|---|
| **Unified RM** | **0.972** | **0.806** | **0.781** | **0.773** | **0.769** |
| **User-based RM** | 1.017 | 0.839 | 0.809 | 0.800 | 0.797 |
| **Item-based RM** | 1.018 | 0.922 | 0.875 | 0.841 | 0.823 |
| - | **P-value** | **P-value** | **P-value** | **P-value** | **P-value** |
| **Unified - UserRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |
| **Unified - ItemRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |

(c) Num. of user rating per item: <50

| **User Sparsity** | **5** | **10** | **20** | **30** | **38** |
|---|---|---|---|---|---|
| **Unified RM** | **0.957** | **0.787** | **0.761** | **0.753** | **0.749** |
| **User-based RM** | 0.987 | 0.809 | 0.780 | 0.772 | 0.769 |
| **Item-based RM** | 0.999 | 0.892 | 0.838 | 0.805 | 0.790 |
| - | **P-value** | **P-value** | **P-value** | **P-value** | **P-value** |
| **Unified - UserRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |
| **Unified - ItemRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |

(d) Num. of user rating per item: <100

item ratings provided by this user is small, it is difficult for the test user to find other users who share ratings among the small amount of item ratings provided. Consequently, the test user has less neighbors, calling for a wide Parzen-window (a larger bandwidth parameter $h_u$) such that the users with relatively large distance can still contribute and smooth the density estimation (rating prediction). However, as the number of item ratings per user increases, for a given test user, he or she has more item ratings to be used to find similar users. In this case, the test user has more neighbors. Thus, it is expected to have a smaller bandwidth parameter to produce a narrow kernel so as to give more emphasis on the most similar users for the density estimation. In both cases, bandwidth parameter $h_i$ varies less than $h_u$, because the item sparsity remains relatively constant. Fig. 10(a) and (b) demonstrate the same behaviour when varying item sparsity. A low number of user ratings per item results in a relatively large bandwidth parameter $h_i$. Conversely, when the number of user ratings per item increases, a small bandwidth parameter $h_i$ is obtained.

Table VI. Comparison among the three derived models on the EachMovie 1 data set. The MAE and P-value of the 10-fold Wilcoxon signed-rank test are reported (the minimum P-value for 10-fold is 0.002).

| **User Sparsity** | **5** | **10** | **20** | **30** | **38** |
|---|---|---|---|---|---|
| **Unified RM** | **1.282** | **1.073** | **1.027** | **1.010** | **1.007** |
| **User-based RM** | 1.353 | 1.156 | 1.102 | 1.087 | 1.087 |
| **Item-based RM** | 1.290 | 1.153 | 1.104 | 1.070 | 1.059 |
| - | **P-value** | **P-value** | **P-value** | **P-value** | **P-value** |
| **Unified - UserRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |
| **Unified - ItemRM** | 0.557(>0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.004(<0.05) | 0.010(<0.05) |

(a) Num. of user rating per item: <20

| **User Sparsity** | **5** | **10** | **20** | **30** | **38** |
|---|---|---|---|---|---|
| **Unified RM** | **1.249** | **1.040** | **0.999** | **0.983** | **0.983** |
| **User-based RM** | 1.338 | 1.128 | 1.076 | 1.062 | 1.060 |
| **Item-based RM** | 1.293 | 1.155 | 1.098 | 1.069 | 1.059 |
| - | **P-value** | **P-value** | **P-value** | **P-value** | **P-value** |
| **Unified - UserRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |
| **Unified - ItemRM** | 0.006(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |

(b) Num. of user rating per item: <30

| **User Sparsity** | **5** | **10** | **20** | **30** | **38** |
|---|---|---|---|---|---|
| **Unified RM** | **1.238** | **1.025** | **0.986** | **0.972** | **0.971** |
| **User-based RM** | 1.326 | 1.095 | 1.049 | 1.028 | 1.028 |
| **Item-based RM** | 1.296 | 1.159 | 1.096 | 1.067 | 1.057 |
| - | **P-value** | **P-value** | **P-value** | **P-value** | **P-value** |
| **Unified - UserRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |
| **Unified - ItemRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |

(c) Num. of user rating per item: <50

| **User Sparsity** | **5** | **10** | **20** | **30** | **38** |
|---|---|---|---|---|---|
| **Unified RM** | **1.202** | **0.973** | **0.934** | **0.922** | **0.920** |
| **User-based RM** | 1.267 | 1.026 | 0.978 | 0.960 | 0.960 |
| **Item-based RM** | 1.263 | 1.098 | 1.033 | 1.006 | 0.999 |
| - | **P-value** | **P-value** | **P-value** | **P-value** | **P-value** |
| **Unified - UserRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |
| **Unified - ItemRM** | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) | 0.002(<0.05) |

(d) Num. of user rating per item: <100

We conclude that the EM algorithm adapts the bandwidth parameters successfully to the different sparsity situations. Let us now compare the performance of the three different models for rating prediction: the unified relevance model, the user-based relevance model and the item-based relevance model (Eq. 40a–40c).

We vary user sparsity at 2, 10, 20, 30 and 38, and item sparsity ranging from <20, <30, <50 to <100. In both the MovieLens 1 and EachMovie 1 data sets, we randomly assign 400 users to the training set, and use the remaining users as the test set. Fig. 11 and 12 summarise the results on the Movielens 1 and the Eachmovie 1 data sets, respectively. The experiments with varying user and item sparsity settings show that the MAE performance of each rating prediction model improves with the number of given ratings per test user. Figures 11(c), 12(c) and (d), demonstrate that the user-based relevance model improves more from a higher number of given ratings per test item than the item-based relevance model does, especially in the EachMovie case. At first sight, this is surprising as we would expect

the item-based relevance model to improve most from a reduced item sparsity (i.e., from having a more reliable item-to-item similarity measure). Careful investigating of this finding shows however that prediction accuracy does not only depend upon the reliability of the similarity measure, but *also* relies on the number of similar ratings that contribute to the predictions. The larger number of given ratings per test item improves the reliability of the item-to-item similarity measure in the item-based relevance model, but it *also* increases the number of ratings by users that are similar to the test users (the SURs) in the user-based relevance model. Both effects contribute to better rating predictions, but increasing the number of SURs proves to be more beneficial.

Fig. 12(d) shows how the user-based relevance model gradually outperforms the item-based relevance model as the number of given user rating per item increases. More importantly however, we find that the unified relevance model outperforms the user-based relevance model and the item-based relevance model in all sparsity settings. Tables V and VI list more details of the performance comparison over the two different data sets, to investigate the statistical significance of the performance improvement obtained by the unified relevance model. It shows the P-value of a Wilcoxon signed-rank test [Hull 1993] applied to each configuration. We conclude that the unified relevance model consistently and significantly improves the recommendation performance over the user-based and item-based relevance models, irrespective of the sparsity (except for the one exception in the top left corner of Table VI, where the difference with the item-based relevance model is not significant). We conclude that the unified relevance model is indeed effective at fusing the predictions from user and item aspects.

5.3.3 *Comparison to other approaches.* We continue with a comparison to results obtained with other methods. Each setting uses the optimal $h_u$ and $h_i$ learned with the EM algorithm.

Recall that our unified model provides a very general framework for collaborative filtering, particularly for those that make use of the neighborhood concept. Thus, we first compare our unified model with other popular methods that need to compute the paired similarities, for instance, the memory-based approaches. Also, the Personality Diagnosis method (see [Pennock et al. 2000]) can be considered as a neighborhood-based approach as it requires to pre-compute the conditional probabilities (similarities) between two paired users. Table VII presents the comparison of our unified model to the the Personality Diagnosis (PD) method and the four memory-based approaches, i.e., the used-based Pearson Correlation Coefficient (UserPCC) and Vector Space (UserVS ) methods (see [Breese et al. 1998]), and the item-based Pearson Correlation Coefficient (ItemPCC) and Vector Space (ItemVS) methods (see [Sarwar et al. 2001]). In addition, we also conduct a significance test, showing the P-value of a Wilcoxon signed-rank test applied to each configuration. From the table, we can see that the recommendation performance of our unified model is indeed significantly better than that of other alternatives that have been considered.

Next we adopt the MovieLens 2 data set [Si and Jin 2003] (called the MovieRating test bed in [Jin et al. 2004; Xue et al. 2005]) as well as the two EachMovie data sets. We followed the evaluation procedure described in [Xue et al. 2005] and [Si

Table VII. Comparison with other approaches on the EachMovie 1 and MovieLens 1 data sets. The MAE and P-value of the 10-fold Wilcoxon signed-rank test are reported (the minimum P-value for 10-fold is 0.002).

| **Given Ratings** | 5 | 10 | 20 | 30 |
|---|---|---|---|---|
| **Unified RM** | 1.011 | 0.919 | 0.887 | 0.876 |
| **UserVS** | 1.108 | 1.041 | 1.023 | 1.018 |
| **UserPCC** | 1.079 | 0.954 | 0.911 | 0.893 |
| **ItemVS** | 1.101 | 1.025 | 0.998 | 0.988 |
| **ItemPCC** | 1.120 | 1.013 | 0.969 | 0.954 |
| **PD** | 1.187 | 1.084 | 1.055 | 1.050 |
| **-** | **P-value** | **P-value** | **P-value** | **P-value** |
| **Unified - UserVS** | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) |
| **Unified - UserPCC** | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) |
| **Unified - ItemVS** | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) |
| **Unified - ItemPCC** | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) |
| **Unified - PD** | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) |

(a) the EachMovie 1 data set.

| **Given Ratings** | 5 | 10 | 20 | 30 |
|---|---|---|---|---|
| **Unified RM** | 0.837 | 0.769 | 0.749 | 0.741 |
| **UserVS** | 0.900 | 0.845 | 0.832 | 0.828 |
| **UserPCC** | 0.888 | 0.803 | 0.775 | 0.762 |
| **ItemVS** | 0.910 | 0.829 | 0.803 | 0.794 |
| **ItemPCC** | 0.954 | 0.865 | 0.813 | 0.795 |
| **PD** | 0.927 | 0.865 | 0.837 | 0.827 |
| **-** | **P-value** | **P-value** | **P-value** | **P-value** |
| **Unified - UserVS** | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) |
| **Unified - UserPCC** | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) |
| **Unified - ItemVS** | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) |
| **Unified - ItemPCC** | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) |
| **Unified - PD** | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) | 0.002 (<0.05) |

(b) the MovieLens 1 data set.

and Jin 2003], aiming to compare the performance of our unified model with the state-of-art results of the mixture models [Si and Jin 2003] and the cluster-based models [Xue et al. 2005]. Table VIII presents the comparison of our experimental results to the six methods of [Si and Jin 2003], i.e., the two extensions of the Aspect Models (AM_c,AM_d, see [Si and Jin 2003]), 'Personality Diagnosis' (PD) ([Pennock et al. 2000]), the user-based Pearson Correlation Coefficient (PCC) ([Breese et al. 1998]), Vector Space (VC), and, Flexible Mixture Model (FMM) ([Si and Jin 2003]). On the Eachmovie 1 data set, our method outperforms all of these methods in all configurations. In the MovieLens 2 data set, only FMM attains comparable results. However the FMM method has more computation complexity than our unified model in the online recommendation phase as it requires the EM iterations called "fold-in" to find both the hidden user clusters and item clusters for new users.

Table IX shows our experimental results as well as the results listed in [Xue et al. 2005], i.e., the cluster-based Pearson Correlation Coefficient (SCBPCC) and the cluster-based collaborative filtering (CBCF) ([Xue et al. 2005]), the Aspect Models (AM) ([Hofmann 2004]), 'Personality Diagnosis' (PD) ([Pennock et al. 2000]), and the user-based Pearson Correlation Coefficient (PCC) and Vector Space (VC) ([Breese et al. 1998]). For both two test sets, our method outperforms these methods in all configurations. By unifying the ratings from both user and item aspects for prediction, our unified relevance model is found to be effective in improving the

Table VIII. Comparison with the result reported in [Si and Jin 2003]. The MAE is reported.

| **Training Users:** | **200** | | | **400** | | |
|---|---|---|---|---|---|---|
| **Given Ratings:** | **5** | **10** | **20** | **5** | **10** | **20** |
| **Unified Model** | **1.05** | **0.97** | **0.94** | **1.04** | **0.96** | **0.93** |
| **PCC** | 1.22 | 1.16 | 1.13 | 1.22 | 1.16 | 1.13 |
| **VS** | 1.25 | 1.24 | 1.26 | 1.32 | 1.33 | 1.37 |
| **PD** | 1.19 | 1.16 | 1.15 | 1.18 | 1.16 | 1.15 |
| **AM_a(20)** | 1.27 | 1.18 | 1.14 | 1.28 | 1.19 | 1.16 |
| **AM_a(10)** | 1.18 | 1.17 | 1.16 | 1.15 | 1.14 | 1.13 |
| **FMM** | 1.07 | 1.04 | 1.02 | 1.05 | 1.03 | 1.01 |

(a) the EachMovie 1 data set

| **Training Users:** | **100** | | | **200** | | |
|---|---|---|---|---|---|---|
| **Given Ratings:** | **5** | **10** | **20** | **5** | **10** | **20** |
| **Unified Model** | 0.848 | **0.779** | **0.796** | 0.828 | **0.767** | 0.781 |
| **PCC** | 0.881 | 0.832 | 0.809 | 0.878 | 0.828 | 0.801 |
| **VS** | 0.859 | 0.834 | 0.823 | 0.862 | 0.950 | 0.854 |
| **PD** | 0.839 | 0.826 | 0.818 | 0.835 | 0.816 | 0.806 |
| **AM_a(5)** | 0.882 | 0.856 | 0.836 | 0.891 | 0.850 | 0.818 |
| **AM_a(2)** | 0.869 | 0.857 | 0.850 | 0.837 | 0.833 | 0.825 |
| **FMM** | **0.829** | 0.822 | 0.807 | **0.800** | 0.787 | **0.768** |

(b) the MovieLens 2 data set

Table IX. Comparison with the results reported in [Xue et al. 2005]. The MAE is reported.

| **Training Users:** | **500** | | | **2000** | | | **6000** | | |
|---|---|---|---|---|---|---|---|---|---|
| **Given Ratings:** | **5** | **10** | **20** | **5** | **10** | **20** | **5** | **10** | **20** |
| **Unified Model** | **1.061** | **0.969** | **0.938** | **1.054** | **0.957** | **0.921** | **1.061** | **0.954** | **0.918** |
| **PCC** | 1.157 | 1.075 | 1.048 | 1.124 | 1.052 | 1.020 | 1.118 | 1.039 | 0.988 |
| **PD** | 1.148 | 1.145 | 1.140 | 1.129 | 1.087 | 1.043 | 1.101 | 1.063 | 1.051 |
| **AM** | 1.157 | 1.082 | 1.057 | 1.125 | 1.078 | 1.054 | 1.117 | 1.069 | 1.046 |
| **CBCF** | 1.207 | 1.132 | 1.089 | 1.187 | 1.113 | 1.063 | 1.197 | 1.111 | 1.060 |
| **SCBPCC** | 1.105 | 1.041 | 1.004 | 1.085 | 1.014 | 0.973 | 1.073 | 1.001 | 0.956 |

(a) the EachMovie 2 data set

| **Training Users:** | **100** | | | **200** | | | **300** | | |
|---|---|---|---|---|---|---|---|---|---|
| **Given Ratings:** | **5** | **10** | **20** | **5** | **10** | **20** | **5** | **10** | **20** |
| **Unified Model** | **0.848** | **0.779** | **0.796** | **0.828** | **0.767** | **0.781** | **0.799** | **0.7552** | **0.764** |
| **PCC** | 0.874 | 0.836 | 0.818 | 0.859 | 0.829 | 0.813 | 0.849 | 0.841 | 0.820 |
| **PD** | 0.849 | 0.817 | 0.808 | 0.836 | 0.815 | 0.792 | 0.827 | 0.815 | 0.789 |
| **AM** | 0.963 | 0.922 | 0.887 | 0.849 | 0.837 | 0.815 | 0.820 | 0.822 | 0.796 |
| **CBCF** | 0.924 | 0.896 | 0.890 | 0.908 | 0.879 | 0.852 | 0.847 | 0.846 | 0.821 |
| **SCBPCC** | 0.848 | 0.819 | 0.789 | 0.831 | 0.813 | 0.784 | 0.822 | 0.810 | 0.778 |

(b) the MovieLens 2 data set

prediction accuracy for recommendation consistently.

## 6. CONCLUSIONS AND FUTURE WORK

This paper presented a unified probabilistic model for collaborative filtering. We explain how to use Parzen-window density estimation for acquiring the probabilities of the proposed unified relevance model. We generalised the kernel density estimation by applying the 'kernel trick', and showed that the often used cosine measure is a suitable projection kernel function. The resulting method has been shown to produce highly accurate predictions on common benchmark data.

The probabilistic framework calls for interesting future work. Firstly, we intend

to explore smoothing techniques as an extra technique to tackle data sparsity. For instance, it is possible to use interpolation smoothing to introduce a background model into the density estimation. Secondly, we plan to look at other IR models for collaborative filtering problems, especially in the situation where we need to pose collaborative filtering as an item ranking problem. The well-known Probability Ranking Principle (PRP) of information retrieval [Robertson 1997] is of particular interest as it provides a theoretical guideline for ranking documents (items). In this regard, we will investigate the possible usages of other ranking models such as the language modelling of information retrieval [Croft and Lafferty 2003; Wang et al. 2006a] and the BM25 ranking formulas [Robertson and Walker 1994]. Thirdly, to deal with different scenarios in recommender systems, we will investigate the possible integration of other text retrieval techniques (more specifically, query expansion and relevance feedback). Fourthly, since our methods are general models for co-occurrence data, it is also worthwhile seeking the possible usage of the models beyond collaborative filtering. We are particularly interested in applying the unified relevance model for the unification of document and query generation in text retrieval.

## REFERENCES


Bodoff, D. 1999. A re-unification of two competing models for document retrieval. *J. Am. Soc. Inf. Sci. 50,* 1, 49–64.

Bodoff, D. and Robertson, S. 2004. A new unified probabilistic model. *J. Am. Soc. Inf. Sci. Technol. 55,* 6, 471–487.

Breese, J., Heckerman, D., and Kadie, C. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI-98).* Morgan Kaufmann, San Francisco, CA, 43–52.

Canny, J. 2002. Collaborative filtering with privacy via factor analysis. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval.* ACM Press, New York, NY, 238–245.

Cheung, K.-W. and Tian, L. F. 2004. Learning user similarity and rating style for collaborative recommendation. *Inf. Retr. 7,* 3-4, 395–410.

Croft, W. B. and Lafferty, J. 2003. *Language Modeling for Information Retrieval.* Kluwer Academic Publishers, Norwell, MA, USA.

DataSet. MovieLens: `http://www.grouplens.org/`.

DataSet. EachMovie: `http://research.compaq.com/SRC/eachmovie/`.

Dempster, A. P., Laird, N. M., and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society 39,* 1, 1–38.

Deshpande, M. and Karypis, G. 2004. Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst. 22,* 1, 143–177.

Duda, R. O., Hart, P. E., and Stork, D. G. 2001. *Pattern Classification.* Wiley Interscience, Wiley, New York.

Duin, R. P. W. 1976. On the choice of smoothing parameters for parzen estimators of probability density functions. *IEEE Transactions on Computers C-25*, 1175– 1179.

Goldberg, K., Roeder, T., Gupta, D., and Perkins, C. 2001. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval Journal 4,* 2 (July), 133–151.

Herlocker, J. L. 2000. Understanding and improving automated collaborative filtering systems. Ph.D. thesis, University of Minnesota. Adviser-Joseph A. Konstan.

Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. 1999. An algorithmic framework for performing collaborative filtering. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval.* ACM Press, New York, NY, 230–237.

Hiemstra, D. 2001. Using language models for information retrieval. Ph.D. thesis, University of Twente.

Hofmann, T. 2004. Latent semantic models for collaborative filtering. *ACM Trans. Info. Syst. Vol 22(1)*, 89–115.

Hu, R. and Lu, Y. 2006. A hybrid user and item-based collaborative filtering with smoothing on sparse data. *icat 0*, 184–189.

Huang, Z., Chen, H., and Zeng, D. 2004. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inf. Syst. 22,* 1, 116–142.

Hull, D. 1993. Using statistical testing in the evaluation of retrieval experiments. In *SIGIR '93: Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM Press, New York, NY, 329–338.

Jin, R., Chai, J. Y., and Si, L. 2004. An automatic weighting scheme for collaborative filtering. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM Press, New York, NY, 337–344.

Lafferty, J. and Zhai, C. 2001. Document language models, query models, and risk minimization for information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM Press, New York, NY, 111–119.

Lafferty, J. and Zhai, C. 2003. Probabilistic relevance models based on document and query generation. *Language Modeling and Information Retrieval, Kluwer International Series on Information Retrieval V.13*, 1–10.

Lavrenko, V. 2004. A generative theory of relevance. Ph.D. thesis, University of Massachusetts Amherst. Director-W. Bruce Croft and Director-James Allan.

Lavrenko, V. and Croft, W. B. 2001. Relevance based language models. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM Press, New York, NY, 120–127.

Linden, G., Smith, B., and York, J. 2003. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing Jan/Feb.*, 76–80.

Liu, Q., Lu, H., and Ma, S. 2004. Improving kernel fisher discriminant analysis for face recognition. *IEEE Trans. Circuits Syst. Video Techn. 14,* 1, 42–49.

Maron, M. E. and Kuhns, J. L. 1960. On relevance, probabilistic indexing and information retrieval. *J. ACM 7,* 3, 216–244.

Paclik, P., Novovicova, J., Pudil, P., and Somol, P. 2000. Road sign classification using laplace kernel classifier. *Pattern Recogn. Lett. 21,* 13-14, 1165–1173.

Pennock, D. M., Horvitz, E., Lawrence, S., and Giles, C. L. 2000. Collaborative filtering by personality diagnosis: A hybrid memory and model-based approach. In *UAI '00: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, 473–480.

Ponte, J. M. and Croft, W. B. 1998. A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM Press, New York, NY, 275–281.

Rennie, J. D. M. and Srebro, N. 2005. Fast maximum margin matrix factorization for collaborative prediction. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*. ACM Press, New York, NY, 713–719.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. 1994. Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*. ACM Press, New York, NY, 175–186.

Robertson, S. 2003. The unified model revisited. In *Mathematical/Formal Methods in IR, Workshop in SIGIR 2003*. ACM Press, New York, NY.

Robertson, S. E. 1997. The probability ranking principle in IR. *Readings in information retrieval*, 281–286.

Robertson, S. E. 2005. On event spaces and probabilistic models in information retrieval. *Information Retrieval 8,* 2, 319 – 329.

Robertson, S. E., Maron, M. E., and Cooper, W. 1982. Probability of relevance: a unification of two competing models for document retrieval. *Information Technology: Research and Development 1,* 1, 1–21.

Robertson, S. E. and SparckJones, K. 1976. Relevance weighting of search terms. *Journal of the American Society for Information Science 27,* 3, 129–46.

Robertson, S. E. and Walker, S. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*. Springer-Verlag New York, Inc., New York, NY, 232–241.

Sarwar, B., Karypis, G., Konstan, J., and Reidl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*. ACM Press, New York, NY, 285–295.

Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. T. 2000. Application of dimensionality reduction in recommender system – a case study. In *Proc. of ACM WebKDD Workshop*. ACM Press, New York, NY.

Schölkopf, B. 2000. The kernel trick for distances. In *NIPS*. MIT Press, Cambridge, MA, 301–307.

Schölkopf, B. and Smola, A. J. 2001. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

Shardanand, U. and Maes, P. 1995. Social information filtering: algorithms for automating "word of mouth". In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co., New York, NY, 210–217.

Si, L. and Jin, R. 2003. Flexible mixture model for collaborative filtering. In *ICML*. AAAI Press, DC, 704–711.

Skurichina, M. 1990. Effect of the kernel function form on the quality of nonparametric parzen window classifier. *Statistical Problems of Control 95*, 216–244.

Tomasi, C. 2004. Estimating gaussian mixture densities with EM : A tutorial. Tech. rep., Duke University. `http://www.cs.duke.edu/courses/spring04/cps196.1/handouts/EM/tomasiEM.pdf`.

van Rijsbergen, C. J. 1979. *Information Retrieval*. Butterworths, London, London, UK.

Wang, J., de Vries, A. P., and Reinders, M. J. 2006a. A user-item relevance model for log-based collaborative filtering. In *Proc. of ECIR06, London, UK*. Springer Berlin / Heidelberg, Berlin, Germany, 37–48.

Wang, J., de Vries, A. P., and Reinders, M. J. T. 2006b. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM Press, New York, NY, 501–508.

Xue, G.-R., Lin, C., Yang, Q., Xi, W., Zeng, H.-J., Yu, Y., and Chen, Z. 2005. Scalable collaborative filtering using cluster-based smoothing. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM Press, New York, NY, 114–121.

Zhai, C. and Lafferty, J. 2001. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM Press, New York, NY, 334–342.

## A. CROSS-VALIDATED EM ALGORITHM

This appendix derives a cross-validated expectation maximisation algorithm to select an optimal value of the smoothing parameters $h_u$ and $h_i$. Of course, this depends for a large part on the data: on the number of data points and their distribution. Other factors of influence are the Parzen window function and the optimality criterion.

The goal is to select $h_u$ and $h_i$ such that they maximise the likelihood function:

$$\begin{aligned}\hat{h}_u, \hat{h}_i &= \arg\max_{h_u,h_i} \prod_{(\mathbf{u},\mathbf{i})\in S_r} P(\mathbf{u},\mathbf{i}|r) \\ &= \arg\max_{h_u,h_i} \prod_{(\mathbf{u},\mathbf{i})\in S_r} \frac{1}{|S_r|} \sum_{(\mathbf{u}',\mathbf{i}')\in S_r} \frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})\end{aligned} \tag{46}$$

It is easy to see that the joint distribution reaches an absolute maximum when $h_u = 0$ and $h_i = 0$. [Duin 1976] has proposed cross-validated maximum likelihood estimation to remove this anomaly,

$$\hat{h}_u, \hat{h}_i = \arg\max_{h_u,h_i} \prod_{(\mathbf{u},\mathbf{i})\in S_r} \frac{1}{|S_r - 1|} \sum_{(\mathbf{u}',\mathbf{i}')\in S_r \wedge (\mathbf{u}'\neq\mathbf{u}) \wedge (\mathbf{i}'\neq\mathbf{i})} \frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i}) \tag{47}$$

This equation can be solved using the iterative expectation maximisation (EM) algorithm (e.g., [Paclik et al. 2000]).

The Parzen-window density estimation method can be interpreted as a generative model with a large mixture of $|S_r - 1|$ (because of cross-validation) component densities with equal weight, where the means of the component densities (assuming a symmetric window function) are located at each observation. Test samples $\mathbf{u}$ and $\mathbf{i}$ are generated from component densities with means $\mathbf{u}'$ and $\mathbf{i}'$, *i.e.*, $P(\mathbf{u},\mathbf{i}|\mathbf{u}',\mathbf{i}' : \mathbf{h_u},\mathbf{h_i})$, with a prior probability of selecting that component equal to $P(\mathbf{u}',\mathbf{i}') = 1/|S_r - 1|$. Applying Bayes' rule to turn $P(\mathbf{u},\mathbf{i}|\mathbf{u}',\mathbf{i}')$ into $P(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})$ gives :

$$\begin{aligned}P(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i}) &= \frac{P(\mathbf{u}',\mathbf{i}')P(\mathbf{u},\mathbf{i}|\mathbf{u}',\mathbf{i}' : \mathbf{h_u},\mathbf{h_i})}{\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P(\mathbf{u}',\mathbf{i}')P(\mathbf{u},\mathbf{i}|\mathbf{u}',\mathbf{i}' : \mathbf{h_u},\mathbf{h_i})} \\ &= \frac{(1/|S_r-1|)\frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})}{\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}}(1/|S_r-1|)\frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})} \\ &= \frac{\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})}{\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})}\end{aligned} \tag{48}$$

Equation 48 gives the E-step of the EM algorithm. The M-step of the EM algorithm uses a lower-bound $\Lambda(h_u, h_i|h_u^{(t)}, h_i^{(t)})$ to approximate the original maximum likelihood function, and then maximises the lower-bound. The E-step and M-step are iteratively applied until the algorithm converges to a (local) maximum [Tomasi 2004]. This lower-bound towards the log form of the cross-validated likelihood function is obtained from *Jensen's Inequality*, which states that for any concave function $\mathbf{f}$, such that:

$$\mathbf{f}(\sum_i p_i x_i) \geq \sum_i p_i \mathbf{f}(x_i),$$

where $\sum_i p_i = 1, p_i \geq 0$ and $x_i \geq 0$. Because the logarithm is concave in the range of $(0, 1]$, Jensen's Inequality can be used to derive a lower bound for the likelihood

function shown in Eq. 47:

$$
\begin{aligned}
&\prod_{\forall(\mathbf{u},\mathbf{i}):(\mathbf{u},\mathbf{i})\in S_r} \frac{1}{|S_r-1|} \sum_{(\mathbf{u}',\mathbf{i}')\in S_r\wedge(\mathbf{u}'\neq\mathbf{u})\wedge(\mathbf{i}'\neq\mathbf{i})} \frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i}) \\
\propto &\sum_{\forall(\mathbf{u},\mathbf{i}):(\mathbf{u},\mathbf{i})\in S_r} \ln\Big(\frac{1}{|S_r-1|} \sum_{(\mathbf{u}',\mathbf{i}')\in S_r\wedge(\mathbf{u}'\neq\mathbf{u})\wedge(\mathbf{i}'\neq\mathbf{i})} \frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})\Big) \\
= &\sum_{\forall(\mathbf{u},\mathbf{i}):(\mathbf{u},\mathbf{i})\in S_r} \ln\Big(\sum_{(\mathbf{u}',\mathbf{i}')\in S_r\wedge(\mathbf{u}'\neq\mathbf{u})\wedge(\mathbf{i}'\neq\mathbf{i})} \frac{1}{|S_r-1|}\frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})\Big) \\
= &\sum_{\forall(\mathbf{u},\mathbf{i}):(\mathbf{u},\mathbf{i})\in S_r} \ln\Big(\sum_{(\mathbf{u}',\mathbf{i}')\in S_r\wedge(\mathbf{u}'\neq\mathbf{u})\wedge(\mathbf{i}'\neq\mathbf{i})} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})\frac{\frac{1}{|S_r-1|}\frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})}{P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})}\Big) \\
\geq &\sum_{\forall(\mathbf{u},\mathbf{i}):(\mathbf{u},\mathbf{i})\in S_r} \sum_{(\mathbf{u}',\mathbf{i}')\in S_r\wedge(\mathbf{u}'\neq\mathbf{u})\wedge(\mathbf{i}'\neq\mathbf{i})} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})\ln\frac{\frac{1}{|S_r-1|}\frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})}{P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})} \\
=&\Lambda(h_u,h_i|h_u^{(t)},h_i^{(t)})
\end{aligned}
\tag{49}
$$

Thus we have the following lower-bound towards the log form of the cross-validated likelihood function:

$$
\begin{aligned}
\Lambda(h_u,h_i|h_u^{(t)},h_i^{(t)}) =& \sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})\ln\frac{\frac{1}{|S_r-1|}\frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})}{P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})} \\
=& \sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})\ln\frac{1}{|S_r-1|}\frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i}) \\
&-\sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})\ln P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})
\end{aligned}
\tag{50}
$$

The last term can be dropped since it is independent of $h_u$ and $h_i$:

$$
\begin{aligned}
h_u^{(t+1)} =& \arg\max_{h_u}\Lambda(h_u,h_i|h_u^{(t)},h_i^{(t)}) \\
=& \arg\max_{h_u}\sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})\ln\frac{1}{|S_r-1|}\frac{1}{h_u^B}\mathbf{K}(\frac{\mathbf{u}-\mathbf{u}'}{h_u})\frac{1}{h_i^A}\mathbf{K}(\frac{\mathbf{i}-\mathbf{i}'}{h_i})
\end{aligned}
\tag{51}
$$

Solve the maximisation problem of Eq. 51 by taking the derivative of $\Lambda$ with respect to $h_u$. In the case of a Gaussian kernel, first convert the product inside the natural logarithm into a sum of $-B\ln h_u$, $-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}$ and a part that does not depend on

$h_u$:

$$\begin{aligned}
&\frac{\partial}{\partial h_u}\Lambda(h_u, h_i|h_u^{(t)}, h_i^{(t)}) \\
=&\frac{\partial}{\partial h_u}\sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})\ln\frac{1}{|S_r-1|}\frac{1}{{h_u}^B}e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}}\frac{1}{{h_i}^A}e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}} \\
=&\sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})\Big(\frac{-B}{h_u}+\frac{||\mathbf{u}-\mathbf{u}'||^2}{h_u^3}\Big) \qquad (52)\\
=&\sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})\frac{-B}{h_u}+\sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})\frac{||\mathbf{u}-\mathbf{u}'||^2}{h_u^3} \\
=&\frac{-B|S_r|}{h_u}+\sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})\frac{||\mathbf{u}-\mathbf{u}'||^2}{h_u^3}=0
\end{aligned}$$

Therefore, we have:

$$h_u^{(t+1)}=\sqrt{\frac{1}{B|S_r|}\sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})(||\mathbf{u}-\mathbf{u}'||^2)} \qquad (53)$$

Similarly, we have for $h_i$:

$$h_i^{(t+1)}=\sqrt{\frac{1}{A|S_r|}\sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})(||\mathbf{i}-\mathbf{i}'||^2)} \qquad (54)$$

In all, we have our cross-validated EM algorithm to estimate the two bandwidth parameters:

—E step:

$$P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})=\frac{e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}}e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}}}{\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}}e^{-\frac{||\mathbf{u}-\mathbf{u}'||^2}{2{h_u}^2}}e^{-\frac{||\mathbf{i}-\mathbf{i}'||^2}{2{h_i}^2}}} \qquad (55a)$$

—M step:

$$h_u^{(t+1)}=\sqrt{\frac{1}{B|S_r|}\sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})(||\mathbf{u}-\mathbf{u}'||^2)} \qquad (55b)$$

$$h_i^{(t+1)}=\sqrt{\frac{1}{A|S_r|}\sum_{\mathbf{u},\mathbf{i}}\sum_{\mathbf{u}'\neq\mathbf{u},\mathbf{i}'\neq\mathbf{i}} P^{(t)}(\mathbf{u}',\mathbf{i}'|\mathbf{u},\mathbf{i})(||\mathbf{i}-\mathbf{i}'||^2)} \qquad (55c)$$

Received December 2006;