Data Visualization Via Collaborative Filtering

Anne-Marie Kermarrec, Afshin Moin

To cite this version:

Anne-Marie Kermarrec, Afshin Moin. Data Visualization Via Collaborative Filtering. [Research Report] 2012, pp.23. hal-00673330

HAL Id: hal-00673330

https://hal.inria.fr/hal-00673330

Submitted on 23 Feb 2012

Data Visualization Via Collaborative Filtering[*]

Anne-Marie Kermarrec and Afshin Moin

INRIA Rennes Bretagne Atlantique, France

Abstract

Collaborative Filtering (CF) is the most successful approach to Recommender Systems (RS). In this paper, we suggest methods for global and personalized visualization of CF data. Users and items are first embedded into a high-dimensional latent feature space according to a predictor function particularly designated to conform with visualization requirements. The data is then projected into 2-dimensional space by Principal Component Analysis (PCA) and Curvilinear Component Analysis (CCA). Each projection technique targets a different application, and has its own advantages. PCA places *all* items on a *Global Item Map (GIM)* such that the correlation between their latent features is revealed optimally. CCA draws *personalized Item Maps (PIMs)* representing a *small subset of items* to a specific user. Unlike in GIM, a user is present in PIM and items are placed closer or further to her based on their predicted ratings. The intra-item semantic correlations are inherited from the high-dimensional space as much as possible. The algorithms are tested on three versions of the MovieLens dataset and the Netflix dataset to show they combine good accuracy with satisfactory visual properties. We rely on a few examples to argue our methods can reveal links which are hard to be extracted, even if explicit item features are available.

1 Introduction

Collaborative Filtering (CF) is the prevalent approach to Recommender Systems (RS). The goal of RSs is to guide users towards their favorite items through the big mass of alternatives available on the Web. They have been for a few years at the center of researchers' attention due to the importance of

[*] This work is supported by the ERC Starting Grant GOSSPLE number 204742.

recommendations in e-commerce, social networking and rating websites like YouTube and Flickr. CF extracts relevant items by tracing the similarities between the past behavior of users. The work on CF boosted following the Netflix Prize competition [3]. The prize was to improve precision of the Netflix recommender system called CineMatch by 10%. In 2009, a union of three teams won the prize. Their approach consisted of a mixture of different CF techniques.

Following the Netflix Prize competition, most works on CF concentrated on improving the prediction precision [18, 22, 24]. Nevertheless, precision is not the only determining factor for the success of RSs. It is also important that results are represented nice and informative as their careless presentation wastes efforts made to enhance precision. Most often, recommendations are presented in the form of a ranked list. Users have multiple interests, and are likely to change their preferences over time [29, 18, 16]. In the same way, one user account may delegate several persons, each have different taste and needs. Though, ranked lists do not unveil the correlation between items unless the user knows them a priori. This issue limits the benefit of recommender systems as it prevents users from making a correct selection based on their current needs.

In this paper, we suggest a Matrix Factorization-like (MF) approach to visualize the results of CF in the form of 2-dimensional maps. This latter is a better alternative to ranked lists as it arranges items based on their similarity. Of course, other solutions like showing the content information of items may also be considered to improve the informativity of recommendations. However, it requires expensive gathering and involved cleaning tasks which companies try to avoid. In addition, interpretation of a visual map is more convenient and quite intuitive. There is an adage saying ”a picture is worth a thousand words”. MF RSs [9, 2, 28] embed users and items in a latent feature space. Missing ratings are then estimated through the inner product of user and item coordinates. MF has been noted on passing to have the property of putting similar items close to each other in the latent feature space [19]. Nevertheless, there are two barriers against their application to draw 2-dimensional maps. First of all, the existing predictor functions are not apt for visualization purposes. Namely, users link intuitively the relevance between users and items to their Euclidean distance on the map. However, the existing predictors, generally variants of the inner product of vectors, need other information (e.x. the angle between user and item coordinates vector). Secondly, MF approaches need more than 2 dimensions to reach their optimal precision. Consequently, proper projection methods must be picked up preserving as much information as possible.

Our approach is composed of two phases: the embedding phase and the projection phase. The first phase embeds users and items into a high-dimensional space based on a predictor aimed at meeting visualization requirements. This predictor is a decreasing function of the Euclidean distance between the corresponding user and item. In the second phase, data is projected from the high-dimensional space into a 2-dimensional map. Two types of maps are presented, each aimed at a different goal. A *global Item Map (GIM)* is laid out using Principal Component Analysis (PCA). GIM shows *the whole set of items* in the 2-dimensional space from the most informative point of view. It is advantageous to view how items are grouped according to the overall opinion of users. Independently of the underlying application, it is a general solution to the visualization of CF data. In order to represent a a *limited number of items* to a specific user, we suggest *personalized Item Maps (PIMs)*. They are interesting in applications like visual recommending. PIMs place items around the active user such that their relevance decreases with their distance from her. They prefer to preserve the *local* structure of the high-dimensional space. In other words, their priority is to preserve short distances rather than long ones. PIMs are generated using Curvilinear Component Analysis (CCA). The original CCA approach is altered in order to give more importance to preservation of distances w.r.t the active user. All PIMs are presented with the same visual parameters such that comparison between PIMs of different users is meaningful.

The precision of our approach is validated running experiments on the three versions of the MovieLens dataset and the Netflix dataset. The concentration in this work is on recommending movies. Despite, the same approach can be used for any other type of data like musics, restaurants, books, etc. CF does not require content information of the items. Despite, we show with a few examples that our maps are very good in unveiling semantic correlations between items.

2 Related Works

The literature related to this work is essentially in two subjects of CF and information visualization. CF is the most popular strategy for recommender systems thanks to advantages like good precision and no need to content information of items. Two common approaches to CF are Neighborhood-based methods [21, 27, 13, 22, 8] and Matrix Factorization (MF) [14, 18, 23, 25, 15]. Both techniques have been widely studied and a rich literature on them is available. Neighborhood-based methods rely on the k Nearest Neighbors (kNN) of an item (item-based approach [21, 22, 8]), or equivalently the k nearest neighbors of a user (user-based approach [13]), to predict missing ratings. The neighborhood is chosen based on a similarity measure like Cosine or Pearson Correlation. Missing ratings are then estimated through a weighted function of neighbors' ratings. Matrix Factorization-based RSs leverage the whole ratings at the same time. They embed users and items in the same latent feature space minimizing a global cost error function. Missing ratings are predicted as the inner product of user and item coordinates vectors.

Visualization of CF has been previously discussed in a few works. In [5] for example the *NEAR* panel is suggested to manage digital collections. A graphical interface is provided to show the latent similarities between items to the end user in a visual manner. This interface is a hierarchical browser, but provides no map. In [20], the authors combine users' listening behavior with social taggings to derive *social audio features* of the songs. Probabilistic Latent Semantic Analysis (PLSA) is consequently applied for dimensionality reduction. In the context of Neighborhood-based CF, classical Multi Dimensional Scaling (MDS) [6, 4] is directly applicable. MDS has been widely used as a visualization technique in many domains including data mining and machine learning. Namely, a symmetric matrix of item-item dissimilarities is computed. A user-specific list of top-k relevant items is then visualized through MDS. This method has the drawback of not mapping a user together with its recommended items, as similarity is only defined between items but not between a user and an item.

Another possible solution to CF visualization is to model data as a bipartite graph with vertices representing users and items, and edges representing the interactions (e.x. [30]). The graph is then visualized by graph drawing algorithms like Spring Embedder [10] or Spring electrical model [11]. Graph drawing algorithms are aimed at drawing readable layouts according to a number of predetermined aesthetic criteria. Consequently, although they lead to visually pleasing results, they do not have convincing predicting ability. In particular, they offer no prediction function to estimate missing ratings.

This work restates the MF approach in the goal of targeting visualizability of the results, while preserving prediction precision. Relating the relevance of users and items to their distance paves the way for usage of dimensionality reduction techniques. Unlike Neighborhood-based visualization through MDS, our approach can project users and items together on the recommendation map. It is also worth noting the conceptual difference between Neighborhood-based and MF approaches. A study of the literature shows the two approaches treat data in different scales [17]. Specifically, MF processes the whole data in a global scale. On the contrary, Neighborhood-based approach concentrates on local similarities inside data. This implies that MF maps and Neighborhood-based maps visualize the correlations inside the data in different scales.

Another asset of this work is aligning the properties of the projection approach with those of the predictor function. Namely, the more items are relevant, the hyperbolically closer to the user they are placed by the predictor function. CCA then captures this property of the predictor by preferring local topology over global topology. As far as we know, there is no prior work adopting such a strategy to CF visualization. Our approach is also a solution to alleviate the *expainability* issues of the MF approach. It is important for a recommender system to be able to explain the reason behind recommendations. Item-based kNN approach has the advantage of good explainability as it recommends based on the items already known to users. On the contrary, explaining the results of MF approach is more challenging, although this latter slightly outperforms its Neighborhood-based counterpart. Visual representation of the MF data provides users with the possibility of choosing an item close to their past favorite items.

3 Matrix Factorization for Visual Prediction

Describing data with predefined features needs content information which in turn requires complicated gathering and cleaning tasks. One way to overcome this problem is to use dimensionality reduction techniques to describe the data by latent features. In machine learning, it is known as Latent Semantic Analysis (LSA). Singular Value Decomposition (SVD) is the most common MF approach to CF. The rating matrix is represented by R, where rows represent users and columns represent items. Each entry r_{ij} is the rating of user i for item j. SVD factors R in the form of $U\Sigma V^T$, where $U_{m\times m}$ and $V_{n\times n}$ are orthonormal matrices, and $\Sigma_{m\times n}$ is a diagonal matrix whose entries are the singular values of R. A low rank approximation matrix $\hat{R}$ is made by picking the largest singular values and corresponding left and right singular vectors. It is proved that SVD provides the best low rank estimation of R according to the Frobenious norm $\left\|R - \hat{R}\right\|_F$. In CF, it is convenient to reformulate the factorization in the form of $R = PQ^T$, where $P = U\Sigma^{-1/2}$ and $Q^T = \Sigma^{-1/2}V^T$. In this case the low rank estimation of P and Q denoted by $\hat{P}$ and $\hat{Q}$ contain user and item latent features.

3.1 Original SVD-like Approach

SVD is computable if all matrix entries are known. In CF however, the majority of ratings are missing. One way to deal with this problem is to fill in the missing entries by imputation. Nevertheless, inexact imputation can decrease the precision of recommendations. On the other hand, it significantly increases the complexity of the computations. A more successful alternative, known as the SVD-like approach, is to estimate the latent features through minimizing a regularized cost function.

$$min \sum_{p*,q*,r*\in R} (r_{ui} - p_u \cdot q_i)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2),$$

where $p_u = (x_u(1), \cdots, x_u(k))$ and $q_i = (y_i(1), \cdots, y_i(k))$ are the latent features of user u and item i in an embedding space of k dimensions. The first term strives to minimize the prediction error over the observed ratings. Since the goal is to predict future ratings, the second terms is added to avoid overfitting the observed ratings. λ is called the regularization factor. This function is minimized using the stochastic gradient descent method. Users and items are moved iteratively by a step length γ in the opposite direction of the gradient until the movement of users and items is negligible. At each iteration, coordinates are updated as:

$$\begin{cases} p_u \leftarrow p_u + \gamma(e_{ui}p_u - \lambda q_i) \\ q_i \leftarrow q_i + \gamma(e_{ui}q_i - \lambda p_u) \end{cases},$$

where $e_{ui} = \hat{r}_{ui} - r_{ui}$ is the prediction error of item i for user u. A local minimum is usually found after few iterations (less than 100). λ and γ are optimized by cross validation on a smaller subset of the ratings.

3.2 Basic Visual Predictor

The original SVD-like approach is not suitable for visualization purposes. The reason is that the inner product of vectors is not natural to be understood by the end user. To bypass this problem, we express our predictor as a function of the Euclidean distance. Let $\hat{r}_{ui} = f(\|p_u - q_i\|)$ be the predicted rating of user u for item i. The cost function has then the following form:

$$min \sum_{p*,q*,r*} (r_{ui} - \hat{r}_{ui})^2 + \lambda(\|p_u\|^2 + \|q_i\|^2).$$

Taking the partial derivatives, the coordinate update equations are obtained:

$$\begin{cases} x_u(j) \leftarrow x_u(j) - \gamma(e_{ui}\frac{x_u(j)-y_i(j)}{\|p_u-q_i\|}\frac{\partial \hat{r}_{ui}}{\partial x_j} + \lambda\frac{x_u(j)}{\|p_u\|}) \\ y_i(j) \leftarrow y_i(j) - \gamma(e_{ui}\frac{y_i(j)-x_u(j)}{\|p_u-q_i\|}\frac{\partial \hat{r_{ui}}}{\partial y_j} + \lambda\frac{y_i(j)}{\|q_i\|}) \end{cases}.$$

We set an inverse relation between the distance of an item from a user and the corresponding rating:

$$\|p_u - q_i\| = \alpha(\frac{1}{r_{ui}} - \beta).$$

The hyperbolic shape of the distance function is particularly useful in CF as the more irrelevant items become w.r.t a user, the faster they get further from her. This property helps us concentrate on visualizing local topology in PIMs while relaxing long distances. Taking the inverse function of the above equation, the objective estimation function is derived:

$$\hat{r}_{ui} = \frac{1}{\frac{1}{\alpha}\|p_u - q_i\| + \beta}. \qquad (1)$$

In this case, partial derivatives are:

$$\frac{\partial \hat{r}_{ui}}{\partial x_i} = \frac{-1}{\alpha\,\|p_u - q_i\|\,(\frac{1}{\alpha}\,\|p_u - q_i\| + \beta)^2}(x_i - y_i),$$
$$\frac{\partial \hat{r}_{ui}}{\partial y_i} = \frac{1}{\alpha\,\|p_u - q_i\|\,(\frac{1}{\alpha}\,\|p_u - q_i\| + \beta)^2}(x_i - y_i).$$

If we define

$$\triangle = \frac{1}{\alpha\,\|p_u - q_i\|\,(\frac{1}{\alpha}\,\|p_u - q_i\| + \beta)^2} = \frac{\hat{r}_{ui}^2}{\alpha\|p_u - q_i\|},$$

coordinates are updated at each iteration as:

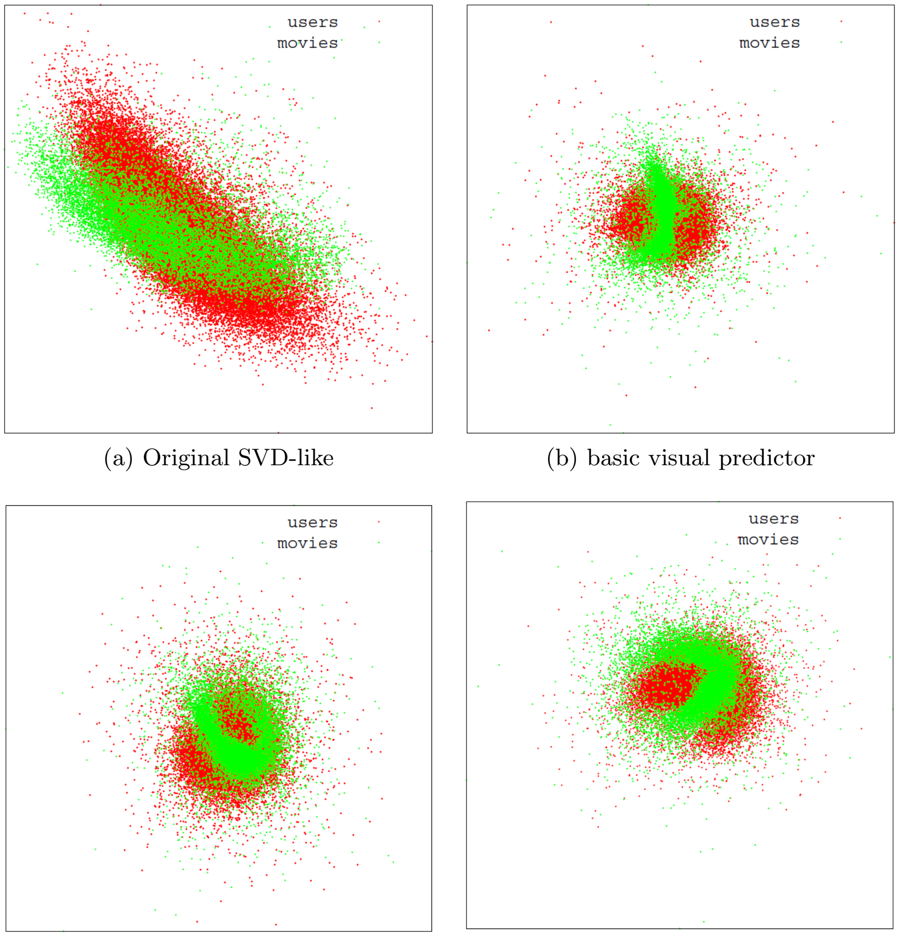
$$\begin{cases} p_u \leftarrow p_u - \gamma(e_{ui}\triangle(q_i - p_u) + \lambda p_u) \\ q_i \leftarrow q_i - \gamma(e_{ui}\triangle(p_u - q_i) + \lambda q_i) \end{cases}, \quad \triangle = \frac{\hat{r}_{ui}^2}{\alpha\|p_u - q_i\|}.$$

α and β are found by cross validation. In theory, other functions varying inversely with the distance may also be used as predictor. We tried hyperbola square and tangent inverse functions. The precision was however worse than the presented predictor. This approach is very similar in the spirit to MF approaches as it embeds users and items in a feature space. Despite, since it is not formulated in form a factorization like SVD, we refer to it as an MF-like approach.

We compare different predictors on the MovieLens and Netflix datasets. To our best knowledge, these are the biggest datasets publically available. MovieLens dataset is provided by GroupLens team in three versions [1]. Each user has rated at least 20 movies. The algorithm is trained on 95% of the data. Predictions are made on the remaining 5%. To form the training and test sets, we have split each user profile into 20 uniform regular slices. 19 slices are chosen as the training profile, while the remaining one serves as the test profile. This way, we assure there is at least one rating in each user's test profile. The Netflix data set was made public for the Netflix Prize competition [3]. The dataset includes $100,480,507$ ratings. $1,480,300$ of the same ratings were provided as the *probe set* for test purposes. We trained our algorithm on all ratings, after having discarded those included in the probe set. The predictions are made on the probe set. Both MovieLens and Netlix are based on a 5-star rating scheme, that is, users rate movies between 1 and 5 stars. While the real ratings are integers, the predictions need not be. In our algorithm, predicted ratings are real numbers. Table 1 summarizes some properties of the four datasets. The quality of the results is compared by the Root Mean Squared Error (RMSE) defined as:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{r_{ui} \in R_T} (\hat{r}_{ui} - r_{ui})^2}{|R_T|}},$$

where R_T is the set of ratings in the test profiles of users. Lower values of RMSE indicate higher prediction precision. For both MoviesLens and Netflix datasets $\alpha = 2.5$ and $\beta = 0.2$ led to good results. For all datasets $\lambda = 0.01$. In the Netflix dataset, $\gamma = 0.005$ for the SVD-like approach and 0.00125 for the visual predictor. In the MovieLens datasets $\gamma = 0.002$ for the SVD-like approach, and 0.0005 for the visual predictor. Precision of the basic visual predictor is shown in Figure 2. It is seen that the precision improves with the number of dimensions up to some threshold. After that, the error increments because the data is not enough to train extra dimensions. In other words, the larger the dataset, the more dimensions can be trained. Figure 1 shows the distribution of MovieLens users and movies in the 2-dimensional space. It is obvious how much the distribution differs from one approach to the other.



(a) Original SVD-like

(b) basic visual predictor

(c) visual Predictor with α_u

(d) improved visual predictor

Figure 1: User and item distribution of different predictors in 2-dimensional projection space.

3.3 Enhancing the Basic Visual Predictor

Users have in general different rating habits to express their opinions about items. For example, they may rate movies with different personal scales, that is, one rates between 1 and 3, while the other rates between 3 and 5. In this case, the same score 3 shows like for the former but dislike for the latter. Consequently, the two users should not be considered similar even if they put the same notes for a number of common items. While a recommender system must be able to handle personal rating habits, the basic visual predictor cannot model it adequately. The reason is that it uses the same model parameters for all users. Figure 1b shows the distribution of users and items in a 2-dimensional embedding space using the basic visual predictor. It is seen that items are spread around a straight line with the end of the line slightly curved to the left. It seems that items can not be moved freely according to the needs of users. As a result, they are put along a line; Users then take some position around the area where the collection of their favorite items is denser. This issue decreases considerably the precision of recommendations. In order to inject more flexibility into the model, we set α_u as a user-specific variable of the model learned automatically from the data. The prediction function is then:

$$\hat{r}_{ui} = \frac{1}{\frac{1}{\alpha_u}\|p_u - q_i\| + \beta}.$$

The regularized cost function is:

$$min \sum_{p*,q*,r*,\alpha*} (r_{ui} - \hat{r}_{ui})^2 + \lambda(\|p_u\|^2 + \|q_i\|^2 + \alpha_u^2).$$

The coordinates are updated as:

$$\begin{cases} p_u \leftarrow p_u - \gamma(e_{ui}\triangle(q_i - p_u) + \lambda p_u) \\ q_i \leftarrow q_i - \gamma(e_{ui}\triangle(p_u - q_i) + \lambda q_i) \\ \alpha_u \leftarrow \alpha_u + \gamma_\alpha(e_{ui}\triangle/\alpha_u - \lambda\alpha_u) \end{cases} , \triangle = \frac{\hat{r}_{ui}^2}{\alpha_u\|p_u - q_i\|}.$$

The values of λ, γ and β are the same as before. We observed that α must be moved with a bigger step length to improve the results. A proper value for all datasets is $\gamma_\alpha = 9\gamma$. The distribution of users and items is seen in Figure 1c. The pattern of items is more twisted this time (see Figure 1b) compared to the basic visual predictor, indicating they have been placed with more freedom. It is seen in Figure 2 that for all datasets precision is greatly improved compared to the basic visual predictor. For the small dataset of MovieLens100K, the visual predictor with user specific α slightly outperforms the SVD-like approach.

3.4 Improved Visual Predictor

The precision of the visual predictor with user-specific α is still worse than the SVD-like approach for larger datasets. To improve further the results, we also set β as a user-specific variable. This helps modeling user behaviors in finer levels, resulting in better positioning of users and items in the embedding space. The prediction function is:

$$\hat{r}_{ui} = \frac{1}{\frac{1}{\alpha_u}\|p_u - q_i\| + \beta_u}.$$

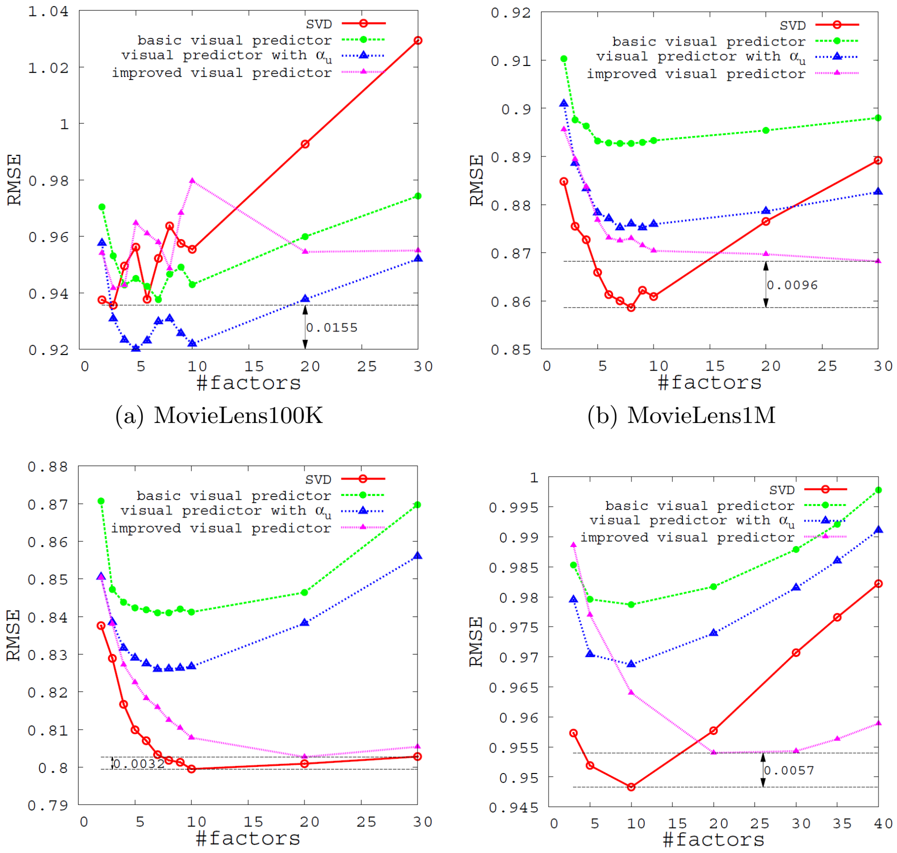
The regularized cost function is:

$$min \sum_{p*,q*,r*,\alpha*,\beta*} (r_{ui} - \hat{r}_{ui})^2 + \lambda(\|p_u\|^2 + \|q_i\|^2 + \alpha_u^2 + \beta_u^2).$$

The coordinates are updated using the following equations:

$$\begin{cases} p_u \leftarrow p_u - \gamma(e_{ui}\triangle(q_i - p_u) + \lambda p_u) \\ q_i \leftarrow q_i - \gamma(e_{ui}\triangle(p_u - q_i) + \lambda q_i) & , \triangle = \frac{\hat{r}_{ui}^2}{\alpha_u\|p_u - q_i\|}. \\ \alpha_u \leftarrow \alpha_u + \gamma_\alpha(e_{ui}\triangle/\alpha_u - \lambda\alpha_u) \\ \beta_u \leftarrow \max\{\beta_u - \gamma_\beta(e_{ui}\hat{r}_{ui}^2 + \lambda\beta_u), \beta_u^{min}\} \end{cases}$$

The values of λ, γ, γ_α are the same as before, and $\gamma_\beta = \gamma/29$. We observed during the experiments that for a few users β_u becomes negative. In order to prevent it, we set a minimum value $\beta_u^{min} > 0$ for β_u. This is a necessary condition for unification of the scale of PIMs in Section 4.2. We set $\beta_u^{min} = 0.05$. The distribution of uses and items is shown in Figure 1d. The item pattern takes the shape of a horseshoe. It is wider and more symmetric than in Figure 1c. It is seen in Figure 2 that precision of the improved visual predictor is considerably better than the visual predictor with user specific α. For the small dataset of MovieLens100K, precision gets worse because the data is not enough to train the new variables. However, interestingly, it is almost as precise as the SVD-like approach for larger datasets. The difference in precision is always less than 0.01, being 0.0096, 0.0032 and 0.0057 for MovieLens1M, MovieLens and Netflix, respectively. This is a good sign as it allows for visual representaion of the results without considerable



(a) MovieLens100K

(b) MovieLens1M

(c) MovieLens

(d) Netflix

Figure 2: RMSE of various approaches on different datasets.

loss in precision. Note it is still possible to improve slightly the precision adding item-specific parameters as in [18] or [15]. However, we avoid doing so because the predictions would not only depend on the active user and her distance from the corresponding item, but also on the properties of the item. Consequently, it is no more certain that items closer to a user have necessarily higher predicted ratings than those further from her.

4 Projection to Lower Dimensions

We embedded users and items into a high-dimensional space because the prediction error is less in higher dimensions. Nevertheless, the embedding

Dataset	users	items	ratings	density%
MovieLens100K	943	1682	100000	6.3
MovieLens1M	6040	3883	1000209	4.24
MovieLens	69878	10681	10000054	1.33
Netflix	480189	17770	100480507	1.18

Table 1: Properties of the datasets

is only half of the problem. Remember the goal was to provide item maps where the relevance between two items or that between a user and a movie can be conjectured visually through their Euclidean distance. Hence, we need proper projection methods to come back to 2 dimensions. We suggest two types of maps: *Global Item Map (GIM)* and *Personalized Item Map (PIM)*. GIM projects the whole set of items indicating the semantic correlation between them. PIM is aimed at representing a small set of items to a particular user such that their semantic correlation and predicted ratings conform to the Euclidean distance between them. In addition, predicted ratings depend inversely on the distance of the corresponding item from the active user.

4.1 Global Item Map

A Global Item Map (GIM) is very helpful to have an estimation of how items are similar or dissimilar according to consumers' point of view. It provides a way for clustering items without having any knowledge about their content. In the high-dimensional embedding space, each dimension is related to some latent feature of the data. The distance of users or items on each dimension is a measure of how similar they are in that feature. One way to visualize the data is to show only two features of the data [19]. However, the information of other features is missed. In addition, features are correlated and may contain redundant information. We use Principal Component Analysis (PCA) to generate a GIM. Data projection using PCA has backgrounds in Multi Dimensional Scaling (MDS) and graph drawing [12]. PCA finds the linear combination of all dimensions capturing as much variance of the data as possible. It omits linear dependencies between dimensions so that axes are uncorrelated in the output space. In other words, PCA represents the data from the most *informative* point of view. In CF, the two axes resulted from PCA are indeed two uncorrelated super features, giving the best explanation about the semantic correlation between items than any other linear combination of the original features.

Algorithm 1 Power iteration algorithm

```
$\epsilon \leftarrow 0.0000001$ %$\epsilon$ is some tolerance.
for i = 1 to k do
  $\hat{u}_i \leftarrow$ random
  $\hat{u}_i \leftarrow \frac{\hat{u}_i}{\|\hat{u}_i\|}$
  $u_i \leftarrow \hat{1}$
  while $u_i^T \hat{u}_i < 1 - \epsilon$ do
    $u_i \leftarrow \hat{u}_i$
    %orthogonalize against previous eigenvectors.
    for j = 1 to i-1 do
      $u_i \leftarrow u_i - (u_i^T u_j) u_j$
    end for
    $\hat{u}_i \leftarrow S u_i$
    $\hat{u}_i \leftarrow \frac{\hat{u}}{\|\hat{u}\|}$
  end while
  % $\hat{u}_i$ are the estimated eigenvectors.
end for
```

Item latent features are kept in $\hat{Q}$. PCA uses the first eigenvectors of the Covariance matrix (those corresponding to the highest eigenvectors) as the principal components. To compute the principal components, $\hat{Q}$ is first centered:

$$\bar{\hat{Q}} = \begin{pmatrix} \bar{q}_1(1) & \cdots & \bar{q}_1(k) \\ \vdots & \ddots & \vdots \\ \bar{q}_n(1) & \cdots & \bar{q}_n(k) \end{pmatrix},$$

where $\bar{q}_i(j) = \hat{q}_i(j) - m_j$, and m_j is the mean of column j. Each column $\bar{q}_i$, representing one data point in the input space, is projected along the principal components:

$$y_i = \bar{q}_i^T u.$$

Writing it in the matricial form, the projection is defined as $Y = \hat{Q}^T u$. Since $\bar{\hat{Q}}$ is centered, Y is also centered. We need to maximize the Covariance of the projection:

$$\max{(Y^T Y)} = \max{(u^T \bar{\hat{Q}} \bar{\hat{Q}}^T u)} = \max{(u^T S u)},$$

where $S = \bar{\hat{Q}} \bar{\hat{Q}}^T$ is the Covariance matrix. This is a standard linear algebra maximization problem whose answer is the eigenvector of S corresponding to the highest eigenvalue. In the same way, second eigenvector may be

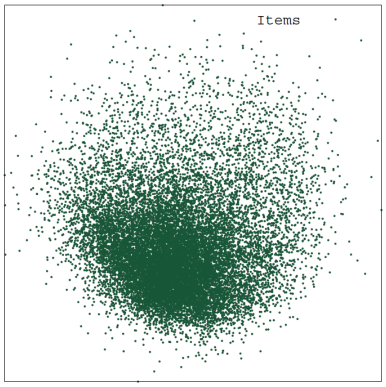


Figure 3: Netflix movies projected from 20d space by PCA

taken as the second dimension. The power iteration algorithm presented in Algorithm 1 can effectively compute the first eignenvetors. Since k is less than 50 in our application, the execution time is negligible.

Figure 3 shows the GIM of Netflix movies projected from a 20-dimensional space. It is seen that the distribution of movies is more uniform than in Figure 1d. This is an advantage of visualization as the drawing surface is used more optimally. Every subset of movies can consequently be picked up for the means of comparison. For example Figure 4 shows 47 movies with more than $140,000$ ratings in the Netflix dataset. Some names are omitted for clarity. With a little knowledge about the movies we see that the two very similar artsy movies *Lost in Translation* and *Royal Tenenbaums* are close to each other while they are far from *Pearl Harbor*, being a patriotic movie with elements of romance and drama. Also the two episodes of *Lord of The Rings* are close to each other. *The day after tomorrow* and *Twister* are both about natural disasters and are on top right of the map. Some continous consistency of movie genre is seen on the map. Specifically, movies on the top left of the map are mostly artsy movies, while those on the top right are actions and advantures. These two groups smoothly join each other in the center just before meeting drama, romance and comedy movies on the bottom.

4.2 Personalized Item Map

Our second objective is to build Personalized Item Maps (PIMs). One possible solution is to take a desired set of items with a user and apply classical MDS (ex. [15]). If the data to be projected contains distances, which is the

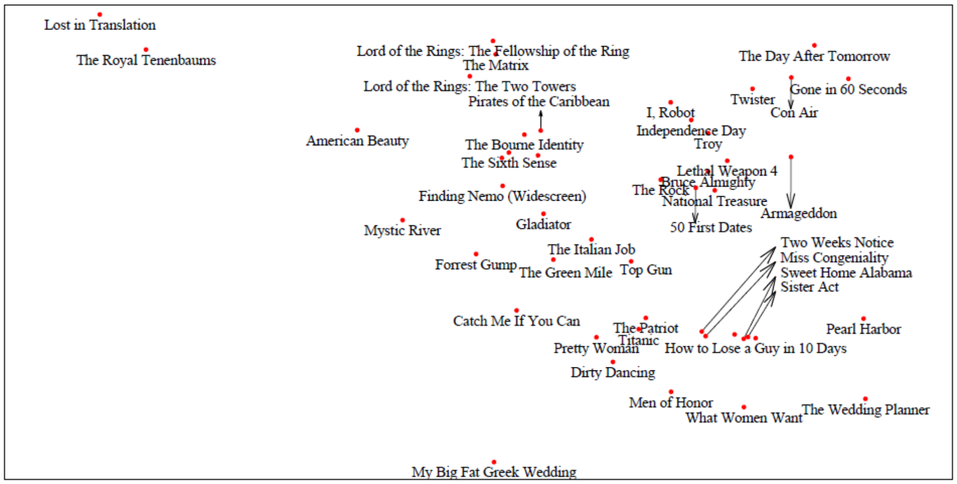


Figure 4: GIM of Netflix movies projected from 20-dimensional space.

case in our application, classical MDS is equivalent to PCA. In CF, when all presented points are of the same type (for example movies), PCA positions pretty well similar points close to each other. This is intuitionally understandable as each new feature recapitulates a combination of the semantics of the input features. Despite, when a set of items is to be represented to a user on a PIM, PCA disturbs so much the original distances that they no more conform to the predicted ratings. Indeed, PCA is a linear mapping preserving best the input distances, but giving the *same* weight to *all* distances. In order to increase the quality of PIMs to an acceptable level, we use nonlinear mapping techniques to prioritize the preservation of shorter distances over longer ones. This idea is also in line with the hyperbolic relation we set in Section 3 between the predicted ratings and the Euclidean distance (between the corresponding user and item), as such relation emphasizes the closeness of relevant items.

Sammon mapping was the first variant of nonlinear maps addressing topology preservation [26]. However, it cannot reproduce all distances, because the output space has in general lower dimensions. Curvilinear Component Analysis (CCA) [7] was suggested as an alternative to Sammon mapping. The idea is to give priority to the preservation of *local* topology. This is done by starting from projecting long distances, and overriding them by shorter ones whenever compromises must be made.

We have slightly altered the original CCA approach to obtain better results in CF. The details are as follows. Suppose we want to show a subset of items together with an active user on a PIM such that first, the semantic correlation between items, represented by their distance, is inherited from the high dimensional embedding space. In the same way, items take distance from the active user based on their irrelevance. Second, preservation of shorter distances is preferred over longer ones. Finally, preserving distances w.r.t the active user is more important as this latter is the central identity to whom the map is presented. We require the distances in the output space to conform with the original spaces as much as possible. The cost function to be minimized is then:

$$E = \frac{1}{2} \sum_{i} \sum_{j \neq i} (X_{ij} - Y_{ij})^2 F(Y_{ij}, \lambda_y),$$

where

$$F(Y_{ij}, \lambda_y) = \begin{cases} 1 \; if \; Y_{ij} \leq \lambda_y \\ 0 \; if \; Y_{ij} > \lambda_y \end{cases} .$$

X_{ij} and Y_{ij} are the distances in the original and the output space, respectively. i and j may refer either to an item or to the active user. We use the principal components as the initial distribution of the points. $F(Y_{ij}, \lambda_y)$ is the step function discarding all distances larger than λ_y in the output space. λy is decreased per cycle in order to prioritize shorter distances in the output space over longer ones.

The cost function can be minimized using the usual stochastic gradient descent approach. However, the complexity of each iteration cycle would be as $O(n^2)$. To decrease the complexity, a different procedure was proposed in [7]. Instead of moving every point according to the influence of all other points, one random point is fixed. Other points are moved with respect to the fixed point, but without consideration of the mutual influence between them. This approach reduces the complexity of each cycle to $O(n)$. The cost may occasionally increase in each cycle, but it decreases in average. If $F(Y_{ij}, \lambda_y)$ is the step function, the movement of each point j w.r.t the fixed point i is:

$$\Delta y_j(i) = \alpha(t) F(Y_{ij}, \lambda_y)(X_{ij} - Y_{ij}) \frac{y_j - y_i}{Y_{ij}} \quad \forall i \neq j.$$

λ_y is decreased per cycle, starting with λ_0 and ending with $\lambda_{t_{max}}$:

$$\lambda_y(t) = \lambda_0 (\frac{\lambda_{t_{max}}}{\lambda_0})^{t/t_{max}}.$$

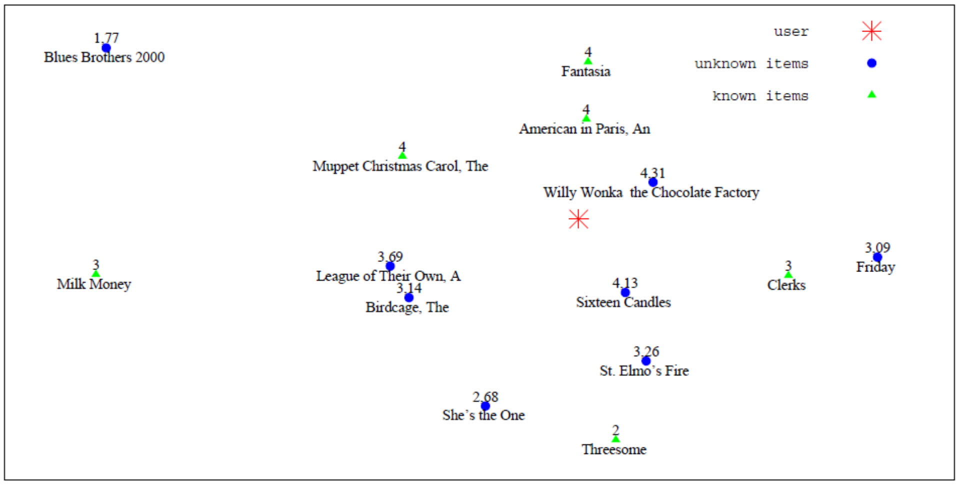


Figure 5: PIM of an anonymous MovieLens user projected from 10-dimensional space.

For each user we set $\lambda_{t_{max}} = \bar{d}_u$, where $\bar{d}_u = \alpha_u(\frac{1}{\bar{r}_u} - \beta_u)$ is the target distance corresponding to the average rating of user u. An item is relevant if its distance from the active user is less than $\bar{d}_u$. Setting $\lambda_{t_{max}} = \bar{d}_u$ ensures that items within some relevant distance from the user on the output space are not discarded until the end of the execution. In our experiments, we set $\lambda_0 = 2.0$. The algorithm is run a number of cycles on the active user and the selected items, denoted by Q_t. In order to give more weight to the preservation of the distances w.r.t the active user, whenever the fixed point is an item, the algorithm is run one extra time on the active user with probability 0.2. This strategy can be validated from a game theoretical point of view. Namely, each point of the data moves the others in the goal of aligning their position in the output space with their original distances from it. Since the algorithm is run in average $0.2\,|Q_t|$ times more on the active user, compromises are made to her benefit.

Projection error of CCA is seen in Figure 6 for an anonymous MovieLens user. Each point shows the distance between two points in the input space dx versus their distance in the output space dy. Closeness to the 45° line indicates better consistency between the original and the output distances. Notice that inconsistencies are less for shorter distances.

Application of user specific parameters to the visual predictor leads to different interpretation of distance in PIMs. It would be nice that all PIMs are represented in the same scale for users can compare their PIMs with each other. To unify the scale of PIMs, we first set the origin to the position of the active user ($p_u = 0$), and reassign all the item coordinates in the translated coordinate system. We look for a transformation such that the predictions remain in the form of Equation (3.2) with the same α and β for all users:

$$\hat{r}_{ui}^{scaled} \quad = \quad \frac{1}{\frac{1}{\alpha}\left\|q_i^{scaled}\right\| + \beta}.$$

Setting $\hat{r}_{ui}^{unscaled} = \hat{r}_{ui}^{scaled}$, we obtain:

$$\left\|q_i^{scaled}\right\| = \frac{\alpha}{\alpha_u}\left\|q_i^{unscaled}\right\| + \alpha(\beta_u - \beta).$$

Since the Euclidean distance is non-negative, $\|q_i^{scaled}\| \geq 0$. This condition is satisfied for all users if $\beta \leq \beta_u^{min}$. The above equation is then held if:

$$q_i^{scaled} = \frac{\alpha}{\alpha_u} q_i^{unscaled} + \frac{q_i^{unscaled}}{\left\|q_i^{unscaled}\right\|}\alpha(\beta_u - \beta).$$

In our experiments we set $\alpha = 2.5$ and $\beta = \beta_u^{min} = 0.05$.

Figure 5 is the PIM of the same user. Shown in the figure, are the movies in her test profile together with a number of movies previously rated by the user. Scores of the known movies are the real ratings of the user. Scores of the test items are the predicted scores, computed in the high dimensional feature space. Note, the predicted rating is a decreasing function of the distance in the embedding space. Hence, the consistency between the predicted ratings and the distance of the user from the corresponding item on PIM is a measure of how much the original distances have been preserved. It is seen that, in general, more relevant items are closer to the user than less relevant ones. The semantic correlation between items is preserved in a very good level. The movies *Friday* and *Clerks* on the right of the user are both comedies. In the same way, *Sixteen Candles*, *St. Elmo's Fire* and *ThreeSome* are movies targeting teenagers with a background of romance and comedy. *An American in Paris* and *Willy Wonka & the Chocolate Factory* are both musicals. This map can help the user choose a movie similar to one of her previous favorites based on her mood in the time of selection. PIMs are also useful in offering *serendipitous* recommendations to a user. Serendipity addresses the diversity of recommended items. Serendipitous recommendations entice users to go through new experiences in their consuming behavior. If the user views a cluster of recommended items on her PIM, none of which is known to her, she is encouraged to go further and try them.

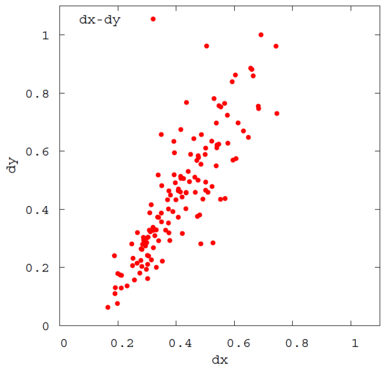


Figure 6: $d_x - d_y$ of the PIM of an anonymous MovieLens user.

5 Discussion

We studied visual representation of CF data. Our approach establishes a link between semantic correlations within data points and their distance in the Euclidean space. Usually, visualization techniques are likely to sacrifice precision for readability. This work presents a unified approach capitalizing on high precision of the MF approach with user-friendly representation of the results. Relating the correlation of data points to their distance in a high-dimensional Euclidean space provides a framework for using the existing projection techniques. We defined our predictor with a hyperbolic function of distance to insist on similarities. Experiments on the three version of MovieLens dataset and the Netflix dataset shows that the visual predictor has almost the same precision as the inner product predictor. Interestingly enough, for the small MovieLens100K ratings dataset, the visual predictor even outperforms the SVD-like algorithm.

A Global Item Map (GIM) was generated using Principal Component Analysis (PCA). It can project the whole set of items at the same time. GIM has a satisfactory performance in regrouping similar items. An altered version of the Curvilinear Component Analysis (CCA) was used to generate Personalized Item Maps (PIMs). PIMs show a small set of items together with the active user. The main difference between PIM and GIM is that *local* topology is prioritized over global topology in PIMs. We also gave more weight to the preservation of distances w.r.t to the active user. All PIMs were reexpressed with the same prediction parameters for intra-user comparison purposes.

The running time of the MF approach is about half an hour on the

Netflix dataset. Fortunately, the algorithm is run offline. It is recomputed with some frequency (for example daily or weekly). On the contrary, running times of projection algorithms is very short. Consequently, they both can be run on demand. However, there is no advantage in generating a GIM more than once after each execution of the MF algorithm as it is a global map.

The magic of this approach is its capability in communicating latent information which are hard to understand even if one is disposed of the content information. Indeed, presented maps are the result of the collective wisdom of a huge number of users, sometimes being more promising than sniffing into tones of expert generated information. The application of the proposed techniques is not limited to recommendations. For example GIM can be used to monitor public opinion about products. Likewise, PIMs can be used to represent a visual comparison between the items of a *digital catalogue*. It is also worth mentioning that although we discussed only item maps, the very same approach can be applied to users. For example a general user map can be generated to detect social groups of consumers showing the same habits in their consumption behavior.

References

[1] *MovieLens Datasets*, 2010. `http://www.grouplens.org/node/73#attachments`.

[2] R. Bell, Y. Koren, and C. Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *ACM SIGKDD international conf. on Knowledge discovery and data mining*, pages 95–104, 2007.

[3] J. Bennett and S. Lanning. The netflix prize. In *KDD Cup and Workshop*, 2007.

[4] I. Borg and P. J. F. Groenen. *Modern Multidimensional Scaling: Theory and Applications (Springer Series in Statistics)*. Springer, 2005.

[5] V. Chen, C. Qian, and R. Woodbury. Visualizing collaborative filtering in digital collections. In *Information Visualization, 2007. IV '07. 11th International Conference*, pages 203 –210, 2007.

[6] T. F. Cox and M. Cox. *Multidimensional Scaling, Second Edition*. Chapman and Hall/CRC, 2000.

[7] P. Demartines and J. Herault. Curvilinear component analysis: a self-organizing neural network for nonlinear mapping of data sets. *Neural Networks, IEEE Transactions on*, pages 148 –154, 1997.

[8] M. Deshpande and G. Karypis. Item based top-n recommendation algorithms. *ACM Transactions on Information Systems*, pages 143–177, 2004.

[9] P. Drineas, I. Kerenidis, and P. Raghavan. Competitive recommendation systems. In *ACM symposium on Theory of computing*, STOC '02, pages 82–90, New York, NY, USA, 2002. ACM.

[10] P. Eades. A heuristic for graph drawing. In *Congressus Numerantium*, pages 149–160, 1984.

[11] T. M. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. In *Software-Practice and Experience*, pages 1129–1164, 1991.

[12] D. Harel and Y. Koren. Graph drawing by high-dimensional embedding. In *Revised Papers from the 10th International Symp. on Graph Drawing*, pages 207–219, 2002.

[13] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *ACM SIGIR*, pages 230–237, 1999.

[14] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115, 2004.

[15] M. Khoshneshin and W. N. Street. Collaborative filtering via euclidean embedding. In *ACM conference on Recommender systems*, RecSys '10, 2010.

[16] J. Z. Kolter and M. A. Maloof. Dynamic weighted majority: An ensemble method for drifting concepts. *J. Mach. Learn. Res.*, pages 2755–2790, 2007.

[17] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proc. of the 14th ACM SIGKDD*, pages 426–434, 2008.

[18] Y. Koren. Collaborative filtering with temporal dynamics. In *Proc. of the 15th ACM SIGKDD*, pages 447–456, 2009.

[19] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, pages 30–37, 2009.

[20] M. Kuhn, R. Wattenhofer, and S. Welten. Social audio features for advanced music retrieval interfaces. In *international conference on Multimedia*, pages 411–420, 2010.

[21] G. Linden, B. Smith, and J. York. Amazon.com recommendations: item-to-item collaborative filtering. In *Internet Computing, IEEE*, pages 76–80, 2003.

[22] R. M. Bell and Y. Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *IEEE International Conference on Data Mining*, pages 43–52, 2007.

[23] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proc. KDD Cup Workshop at SIGKDD'07*, pages 39–42, 2007.

[24] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, 2007.

[25] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted boltzmann machines for collaborative filtering. In *Proc. of the 24th international conference on Machine learning*, pages 791–798, 2007.

[26] J. W. Sammon. A nonlinear mapping for data structure analysis. *IEEE Trans. Comput.*, pages 401–409, 1969.

[27] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl. Item-based collaborative filtering recommendation algorithms. In *World Wide Web*, pages 285–295, 2001.

[28] G. Takács, I. Pilászy, B. Németh, and D. Tikk. Scalable collaborative filtering approaches for large recommender systems. *J. Mach. Learn. Res.*, pages 623–656, 2009.

[29] G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Mach. Learn.*, pages 69–101, 1996.

[30] R. Xiong, M. A. Smith, and S. M. Drucker. Visualizations of collaborative information for end-users. Technical report, 1999.