# Collaborative Filtering through SVD-based and Hierarchical Nonlinear PCA

Manolis G. Vozalis, Angelos Markos, and Konstantinos G. Margaritis

Department of Applied Informatics, University of Macedonia,
156 Egnatia Street, P.O. Box 1591, 54006, Thessaloniki, Greece
{mans,amarkos,kmarg}@uom.gr,

**Abstract.** In this paper, we describe and compare two distinct algorithms aiming at the low-rank approximation of a user-item ratings matrix in the context of Collaborative Filtering (CF). The first one implements standard Principal Component Analysis (PCA) of an association matrix formed from the original data. The second algorithm is based on h-NLPCA, a nonlinear generalization of standard PCA, which utilizes an autoassociative network, and constrains the nonlinear components to have the same hierarchical order as the linear components in standard PCA. We examine the impact of the aforementioned approaches on the quality of the generated predictions through a series of experiments. Experimental results show that the latter approach outperforms the standard PCA approach for most values of the retained dimensions.

**Key words:** Collaborative Filtering,Low-rank Approximation, Artificial Neural Networks, Principal Component Analysis

## 1 Introduction

With the term Collaborative Filtering (CF) we refer to intelligent techniques which are employed by Recommender Systems (RSs) and are used to generate personalized recommendations. The basic idea of CF is that users who have agreed in the past tend to agree in the future. A common and successful approach to collaborative prediction is to fit a factor model to the original rating data, and use it in order to make further predictions. A factor model approximates the observed user preferences in a low dimensionality space in order to uncover latent features that explain user preferences. In this paper, we will focus on two PCA implementations, aiming at the low-rank approximation of the corresponding user-item ratings matrix.

PCA is a well-established data analysis technique that relies on a simple transformation of recorded observations, to produce statistically independent score variables. It has been extensively used for lossy data compression, feature extraction, data visualization, and most recently in the field of Collaborative Filtering [1–3]. The linear assumption underlying PCA makes it insufficient for capturing nonlinear patterns among variables. Artificial Neural Network (ANN) models, a class of nonlinear empirical modeling methods, allow for nonlinear

mappings between the original and the reduced dimensional spaces. Various ANN methods have been described in a PCA framewok [4–7]. ANNs have been utilized for the generation of CF predictions: Billsus and Pazzani [8] formulate CF as a classification problem by feeding their data matrix of reduced dimensions to an ANN. Lee et al. [9] put users into clusters by using a Self-Organizing Map neural network, and then apply CF on those clusters in order to extract recommendations. Gong and Ye [10] utilize a backpropagation neural network to fill the missing values of the original data matrix, and then apply item-based CF to form the item neighborhood.

The aim of this paper is to examine two implementations of PCA in the context of CF. The first implementation utilizes PCA through the Singular Value Decomposition (SVD) of the covariance matrix. For our second implementation we apply a hierarchical nonlinear PCA algorithm, denoted as h-NLPCA [11]. The primary contribution of this work lies in the application of h-NLPCA, which is based on a multi-layer perceptron with an auto-associative topology, for the generation of personalized recommendations. The main advantage of h-NLPCA is that it enforces a hierarchical order of principal components which always yields the same solution of uncorrelated features.

The remainder of this paper is organized as follows: Section 2 is devoted to a general presentation of the two PCA approaches, through SVD and ANNs, respectively. Section 3 discusses the proposed algorithms in the context of CF, outlining the distinct implementation steps. The efficiency of each approach is demonstrated in Section 4 through a set of experiments on a publicly available data set. The paper concludes in Section 5.

## 2 Two PCA Implementations

### 2.1 SVD-based PCA

PCA summarizes the variation in correlated multivariate attributes to a set of non-correlated components, called principal components, each of which is a particular linear combination of the original variables [1]. PCA can be performed by applying the SVD to either a covariance or a correlation matrix of the original data set, in order to extract the smallest number of components while retaining as much of the original variance as possible. The eigenvalues of the covariance (correlation) matrix indicate the amount of variance along the direction given by the corresponding eigenvector. That is, when a covariance matrix $\mathbf{A}$ is decomposed by SVD, i.e., $\mathbf{A} = \mathbf{USV}^T$, the matrix $\mathbf{U}$ contains the variables' loadings for the principal components, and the matrix $\mathbf{S}$ has the corresponding variances along the diagonal [1]. A reduction to $k$ dimensions is obtained by projecting the original data matrix on the subspace consisting of eigenvectors corresponding to the largest $k$ eigenvalues of the covariance matrix.

### 2.2 h-NLPCA

Nonlinear PCA is based on a multi-layer perceptron (MLP) with an autoassociative topology, also known as an *autoencoder*. The network consists of two parts:

the first part represents the extraction function, $\Phi_{extr} : X \rightarrow Z$. The second part represents the inverse, reconstruction function, $\Phi_{gen} : Z \rightarrow \hat{X}$. A hidden layer in each part enables the network to perform nonlinear mapping functions [11]. The autoassociative network performs an identity mapping, which means that the input is approximated at the output layer with the highest possible accuracy. This property of the network is achieved by minimizing the squared reconstruction error $E = \frac{1}{2}||\hat{x} - x||^2$. This task, which is nontrivial, is accomplished by a 'bottleneck' layer in the middle, of smaller dimension than either the input or output layers. Thus, the data have to be projected or compressed into a lower dimensional representation $Z$, for the subsequent layers to reconstruct the input. If network training succeeds in finding an acceptable solution, we may assume that data compression achieved at the 'bottleneck' layer may force hidden units to represent significant features in data.

Hierarchical nonlinear PCA (h-NLPCA), as proposed by Scholz et al. [11], provides the optimal nonlinear subspace spanned by components, but also constrains the nonlinear components to have the same hierarchical order as the linear components in standard PCA. This means that the first $n$ components explain the maximal variance that can be covered by a $n$-dimensional subspace and that the $i$-th component of an $n$ component solution is identical to the $i$-th component of an $m$ component solution. $E_1$ and $E_{1,2}$ are the squared reconstruction errors when using one or two components in the 'bottleneck' layer, respectively. In order to perform the h-NLPCA, we have to minimize both $E_{1,2}$ (as in plain NLPCA, or s-NLPCA), and $E_1$. In practice, this is equal to minimizing the hierarchical error, $E_H$:$E_H = E_1 + E_{1,2}$. The optimal network weights for a minimal error in h-NLPCA can be found by using the conjugate gradient descent algorithm [31]. At each algorithm's iteration, the single error terms $E_1$ and $E_{1,2}$ have to be calculated separately. In standard s-NLPCA, this is performed by a network with either one or two units in the 'bottleneck' layer. In the case of h-NLPCA, one network is the subnetwork of the other. The hierarchical error function can be easily extended to $k$ components ($k \leq d$): $E_H = E_1 + E_{1,2} + E_{1,2,3} + \cdots + E_{1,2,3,\ldots,k}$.

In other words, for the minimization of $E_H$, we search for a $k$-dimensional subspace of minimal mean square error (MSE) under the constraint that the $(k-1)$-dimensional subspace is also of minimal MSE. This requirement is extended so that all $1, \ldots, k$ dimensional subspaces are of minimal MSE. Hence, each subspace represents the data with regard to its dimensionalities best. Hierarchical nonlinear PCA can therefore be seen as a true and natural nonlinear extension of standard linear PCA [11].

## 3 The Proposed Algorithms

In this section we will describe how the aforementioned PCA implementations can be combined with CF in order to make prediction generation both scalable and effective. In both cases, once PCA is applied for the low rank approximation of the original user-item ratings matrix, we compute a neighborhood for

each user. Finally, user similarity is utilized for the generation of the requested prediction.

### 3.1 CF through h-NLPCA

We start with the following basic definitions. For $i = 1, \ldots, n$ users, ratings on $j = 1, \ldots, m$ items are collected in the $n \times m$ data matrix $\mathbf{R}$. Each of the corresponding items takes on $k$ different rating values (levels or categories) from a given range, i.e. $(1, 2, 3, 4, 5)$.
*Step 1. Data representation.* Impute the missing values in the original user-item matrix, $\mathbf{R}$, with the corresponding *column* average, $\bar{r}_j$, which leads to a new filled-in matrix, $\mathbf{A}$.
*Step 2. Low rank approximation.* The conjugate gradient descent algorithm [11] is used to train the h-NLPCA network as described in Section 2. The hierarchical error $E_h$ is minimized at each training iteration. The reduced or reconstructed matrix is denoted as $\mathbf{A}_k$, where $k$ is the number of retained components.
*Step 3. Neighborhood Formation.* Calculate the similarity measure between each user and his closest neighbors in order to form the user neighborhood. To find the proximity between two users, $u_a$ and $u_i$, we utilize the Pearson correlation coefficient, which is computed as follows:

$$cor_{ai} = \frac{\sum_{j=1}^{l} r_{aj} r_{ij}}{\sqrt{\sum_{j=1}^{l} r_{aj} \sum_{j=1}^{h} r_{ij}}}$$

where $r_{ij}$ denotes the rating of user $u_i$ on item $i_j$. Note that the summations over $j$ are calculated over the $l$ items for which both users $u_a$ and $u_i$ have expressed their opinions.
*Step 4. Prediction Generation.* Prediction generation requires that a user neighborhood of size $h$ is already formed for the active user, $u_a$. Then, we compute the prediction rating $p_{aj}$ for user $u_a$ on item $i_j$, using the following equation:

$$p_{aj} = \bar{r_j} + \frac{\sum_{i=1}^{h} rr_{ij} * cor_{ai}}{\sum_{i=1}^{h} |cor_{ai}|}$$

It is important to note that the user ratings, $rr_{ij}$, are taken from the reduced matrix $\mathbf{A}_k$. Also, we have to add the original item average back, $\bar{r_j}$, since it was subtracted during the normalization step of the preprocessing.

### 3.2 CF through SVD-based PCA

*Step 1. Data representation and normalization.* Impute the missing values in the original user-item matrix, $\mathbf{R}$, with the corresponding *column* average, $\bar{r}_j$. Then obtain the column centered matrix $\mathbf{A}$.
*Step 2. Low rank approximation.* Compute the SVD of $\mathbf{A}$ and keep only the first $k$ eigenvalues. This is equivalent to the factorization of the covariance matrix

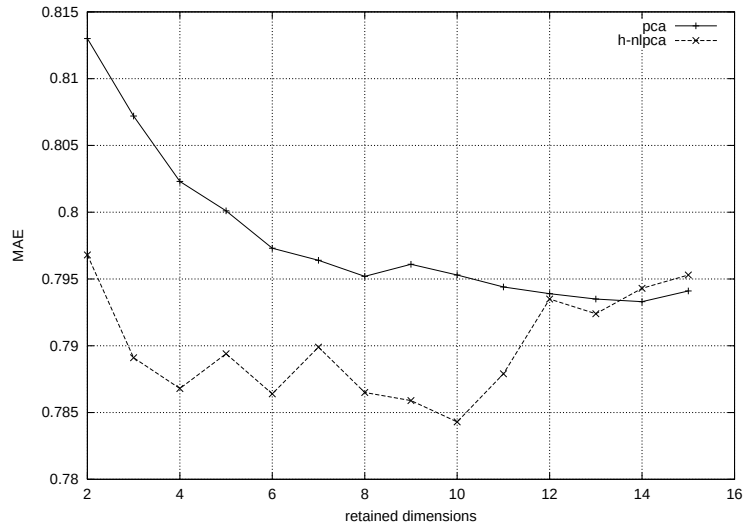


**Fig. 1.** Comparison of PCA and h-NLPCA for different values of retained dimensions

$\frac{1}{m-1}\mathbf{A^T A}$ [1]. The reduced or reconstructed matrix is denoted as $\mathbf{A}_k$, where $k$ is the number of retained components.
Neighborhood formation and prediction generation (steps 3 and 4) are executed as described in the h-NLPCA implementation.

## 4 Experiments

In this section the efficiency of each approach is demonstrated through a series of experiments. We utilized MovieLens, a dataset publicly available from the GroupLens research group, which consists of 100,000 ratings, assigned by 943 users on 1682 movies. The sparsity of the data set is high, at a value of 93.7%. Starting from the initial data set, a distinct split of training (80%) and test (20%) data was utilized. Mean Absolute Error (MAE) was the metric we employed to evaluate the accuracy of the methods. MAE measures the deviation of predictions generated by the RSs from the true rating values, as they were specified by the user.

For our experiments, we kept a fixed user neighborhood size and evaluated the effect of a varying number of retained dimensions, $k$, on prediction accuracy. Figure 1 depicts the MAE for values of $k$ ranging between 2 and 15. Based on that figure, it is clear that h-NLPCA outperformed SVD-based PCA for almost all the values of retained dimensions. In particular, h-NLPCA generated the overall most accurate prediction, MAE=0.7843, for $k$=10, meaning that only 10 *pseudo*-items, out of the 1682 original ones, were able to capture the latent relations existing in the initial user-item ratings matrix. In contrast, SVD-based

PCA reached its lowest error value, MAE=0.7933, for a slightly larger value of $k$, $k$=14.

## 5 Conclusions and Future Work

In this paper we described two factor models, SVD-based PCA and h-NLPCA for the low-rank approximation of a user-item ratings matrix in the context of CF. h-NLPCA, which can be considered as a neural based non-linear extension of PCA, gave the most accurate predictions according to MAE when applied to the MovieLens dataset. The main advantage of the proposed approach stems from the fact that h-NLPCA is able to account for more of the variance in the data compared to SVD-based PCA, when the variables are (or may be) nonlinearly related to each other. However, the prediction accuracy of a certain method depends on the structure of the data it is used on. A detailed comparison on different data sets is beyond the scope of this article. In both PCA implementations, the sparse user-item ratings matrix is filled using the average ratings for users to capture a meaningful latent relationship. Future considerations include PCA methods that are robust against missing data and that allow for missing value estimation. For example, non linear PCA approaches, such as Kernel PCA and Regularized PCA [1], may provide a valuable insight into the CF framework.

## References

1. Jolliffe, I.T.: Principal Component Analysis (2nd Ed.). Springer (2002)
2. Goldberg, K., Roeder, T., Gupta, D., Perkins, C.: Eigentaste: A constant time collaborative filtering algorithm. Information Retrieval Journal **4** (2001) 133–151
3. Kim, D., Yum, B.J.: Collaborative filtering based on iterative principal component analysis. Expert Systems with Applications **28** (May 2005) 823–830
4. Oja, E.: A simplified neuron model as a principal component analyzer. Journal of Mathematical Biology **15**(3) (November 1982) 267–273
5. Diamantaras, K.I., Kung, S.Y.: Principal Component Neural Networks: Theory and Applications. John Wiley & Sons, New York (1996)
6. Kramer, M.A.: Nonlinear principal component analysis using autoassociative neural networks. AIChE Journal **37**(2) (February 1991) 233–243
7. Tan, S., Mayrovouniotis, M.L.: Reducing data dimensionality through optimizing neural network inputs. AIChE Journal **41**(6) (1995) 1471–1480
8. Billsus, D., Pazzani, M.J.: Learning collaborative information filters. In: 15th International Conference on Machine Learning, Madison, WI (1998) 46–53
9. Lee, M., Choi, P., Woo, Y.: A hybrid recommender system combining collaborative filtering with neural network. In Bra, P.D., Brusilovsky, P., Conejo, R., eds.: Adaptive Hypermedia and Adaptive Web-Based Systems. Springer-Verlag, Berlin, Heidelberg (2002) 531–534
10. Gong, S., Ye, H.: An item based collaborative filtering using bp neural networks prediction. In: 2009 International Conference on Industrial and Information Systems, Haikou, China (2009) 146–148
11. Scholz, M., Fraunholz, M., Selbig, J.: Nonlinear principal component analysis: Neural network models and applications. In: Principal Manifolds for Data Visualization and Dimension Reduction. Springer Berlin Heidelberg (2007) 44–67