Personalized TV Recommendation with Mixture Probabilistic Matrix Factorization

Huayu Li * Hengshu Zhu[+] Yong Ge[‡] Yanjie Fu[†] Yuan Ge[±]

Abstract

With the rapid development of smart TV industry, a large number of TV programs have been available for meeting various user interests, which consequently raise a great demand of building personalized TV recommender systems. Indeed, a personalized TV recommender system can greatly help users to obtain their preferred programs and assist TV and channel providers to attract more audiences. While different methods have been proposed for TV recommendations, most of them neglect the mixture of watching groups behind an individual TV. In other words, there may be different groups of audiences at different times in front of a TV. For instance, watching groups of a TV may consist of children, wife and husband, husband, wife, etc in many US household. To this end, in this paper, we propose a Mixture Probabilistic Matrix Factorization (mPMF) model to learn the program preferences of televisions, which assumes that the preference of a given television can be regarded as the mixed preference of different watching groups. Specifically, the latent vector of a television is drawn from a mixture of Gaussian and the mixture number is the estimated number of watching groups behind the television. To evaluate the proposed mPMF model, we conduct extensive experiments with many state-of-the-art baseline methods and evaluation metrics on a real-world data set. The experimental results clearly demonstrate the effectiveness of our model.

Keywords: Smart TV, Recommender Systems, Mixture Probabilistic Matrix Factorization

1 Introduction

Recent years have witnessed the rapid prevalence of smart TV, which can significantly improve users' watching experience by providing a large number of TV programs. Different from traditional televisions, smart TV can capture rich user interactions (e.g., watching behaviors) and record them as device logs for data analysis, which opens a new avenue for building personalized TV recommender systems. Indeed, accurate and personalized TV program recommendation is a crucial demand in smart TV industry. First, given the massive TV programs, it is very difficult for users to find their preferred ones in an efficient way. A personalized TV recommender system would help users easily find relevant programs without spending much time on searching manually. Also, it is very challenging for TV providers (e.g., Google TV and Samsung TV) to deliver relevant content to various users. A personalized TV recommender system will be able to help them attract more audiences with relevant content and consequently earn huge benefit, such as Ads revenue.

While personalized TV recommender systems could greatly benefit both users and TV providers, it is still a very challenging problem mainly due to two following factors. First, there may be different watching groups at different times behind an individual TV in many US households. For instance, watching groups of TV in one household may consist of children, wife and husband, and all family members. These three watching groups have different preference for different TV programs. In other words, the program preference of a television is actually the mixed preference of different watching groups. But, the log recorded at an individual TV includes all programs watched at this TV, thus it is very critical to infer different watching groups behind a single TV and learn their different preference. Second, there are only some implicit feedbacks from users (e.g., watching time) available for developing personalized TV recommender systems. Therefore, it is hard for us to know whether users like a program or not explicitly and how much users like this program based on this kind of implicit data. For example, a long watching time for a program at one TV does not necessarily indicate a user likes it, because it is possible that the program is playing but no one is watching it.

In the literature, some related works on TV recommendations have been reported [5, 19, 7, 13]. For example, Hu *et al* [5] proposed a method based on ma-

*‡Computer Science Department, UNC Charlotte. Email: {hli38,yong.ge}@uncc.edu, [+] Baidu Research-Big Data Lab, zhuhengshu@baidu.com, [†] Rutgers University, yanjie.fu@rutgers.edu, [±] Anhui Polytechnic University, ygetoby@mail.ustc.edu.cn.

trix factorization [14] for TV recommendations, which considers the implicit user behavior as user's preference and the confidence level is proposed to handle such kind of TV data. Xin *et al* [19] proposed an algorithm for modeling TV data with multiple ratings for a same program due to a TV show likely consisting of different episodes. However, these methods neglect the mixture of watching groups and could not effectively uncover the mixed preference behind an individual TV. To address this limitation, in this paper, we propose a novel model to produce more accurate recommendations of TV programs. In general, we assume that there are one or multiple watching groups behind an individual TV. A TV program is watched by a particular watching group when it is played at a television. The preference of a television is decomposed into a mixture preference of its corresponding hidden watching groups who are watching TV individually or together.

Specifically, we propose a two-stage framework for building personalized TV recommender systems. In the first stage, we aim to automatically estimate the number of hidden watching groups for televisions. Due to such large number of televisions, we propose to first cluster televisions into several groups, then estimate the number of watching groups for each cluster separately. In each television group, we perform Markov clustering [16] to learn the number of watching groups, where all the televisions in the same group have the same composition of watching groups. In the second stage, a Mixture Probabilistic Matrix Factorization (mPMF) model is developed to learn the latent vectors of television and program. More specifically, the latent vector of a TV is assumed to be drawn from a mixture of Gaussian where the mixture number is the number of its corresponding watching groups. And all televisions in the same group will share the same Gaussian components but with different coefficient. To evaluate the proposed mPMF model, we conduct extensive experiments with many state-of-the-art baseline methods and validation metrics on a real-world data set. The experimental results clearly demostrate the effectiveness of our model.

2 Personalized TV Recommendation

In this section, we first briefly introduce the real-world data set used in paper, and then propose our method.

2.1 Data Description

The data set used in this paper includes a large number of television playing records that cover a wide range of TV programs. There is various information about each TV program. Specifically, each record contains a television ID, a program ID, the starting time when the program is played at the television, and the time duration that the program is watched at a TV. The information about TV program includes the title of program, and two types of genres/categories of program, namely first-level genre and sub-genre. TV programs include news, TV series, TV shows, and movies. And each program may be played repeatedly. Note that we have only observed TV programs that are played at different televisions, but may be watched by different users. A television represents the device that plays programs and usually is placed in a household or a public place. A user refers to the person who is watching TV programs in front of the television. Users who have similar preferences for TV programs form a watching group.

2.2 Overview of our Method

Given a set of watching records, we focus on producing accurate and personalized recommendations of TV programs for each individual television. TV programs played at a television may be watched by different watching groups at different times. For example, in a family with a couple, they may watch TV programs individually or together. There may be three different watching groups (i.e., wife, husband, couple) in front of TV at different times that have different preference for different TV programs. Therefore, the preference of a television could be decomposed into several sub-preferences of watching groups. To this end, we propose a two-stage framework to build personalized TV recommender systems. Specifically, we first estimate the number of hidden watching groups for each television by clustering watching records. In the second stage, we learn the mixture preference of TV for programs upon the discovered watching groups. In addition, different from many traditional recommender systems, we only have implicit ratings (i.e., watching time of TV programs). Therefore, we also investigate various approaches for addressing implicit ratings in a thorough way and reveal their performances on TV recommendations.

2.3 Discovery of Watching Groups

As discussed above, TV programs played at a television are actually watched by one or multiple watching groups. Particularly for televisions in households, watching groups are usually fixed. Although some works [12, 1, 10, 20] have proposed to analyze user's behaviors for (IP)TV, we will employ a more simple yet effective way to estimate the watching groups for each individual television based on its historical playing records with clustering techniques. In the real-world scenario, there may be millions of televisions, which makes it very inefficient to perform clustering for individual television.

At the same time, watching records of an individual television are usually very sparse and insufficient. Furthermore, we observe that many televisions might have similar composition of watching groups who share similar preferences on TV programs. Therefore, we propose to first cluster televisions into several groups, where similar TV programs are watched in each cluster. Then we estimate the number of watching groups for each cluster.

Television Clustering. In the first step, learning televisions groups in fact can be regarded as a kind of dimensionality reduction. Specifically, the similarity between televisions can be measured by the preference on TV programs. We perform K-means algorithm to cluster televisions by using the watching frequencies of TV programs as features, where the similarity measurement is based on Cosine distance or Euclidean distance. In our data set, there are over 4,000 programs and over 200,000 televisions, thus K-means algorithm on such high-dimensional data is implemented in hierarchical-style method and accelerated by pre-calculating some fixed variables in the distance formula or using some proposed accelerating algorithms [2].

Estimating Watching Groups. After obtaining a set of clusters, the goal of the second step is to learn the number of watching groups in each cluster based on the historical watching records. Indeed, a TV program is watched by a user mainly depending on whether he/she likes its genre and whether he/she has time to watch it. Thus, we propose to utilize Markov clustering [16] to automatically learn the cluster number by considering four types of features: 1) the first-level genre; 2) the sub-genre; 2) the absolute time when a program is watched in a day; and 4) an indicator indicating the watching time is in week day or weekend. As the third feature is represented in a numerical value, we use $1 - \frac{|T_1 - T_2|}{24}$ as its similarity, where T_1 and T_2 are the start playing times in any two records. Other three features are binary and the similarities are calculated with Cosine similarity measurement. Finally, Markov clustering algorithm takes the weighted sum of these four similarities as input to automatically learn the number of clusters, which is regarded as the number of watching groups for each cluster of televisions.

To facilitate the following introduction, we use L to denote the number of television groups/clusters, where the l-th group contains N_l televisions. Meanwhile, programs played at each television in the l-th group are watched by K_l different watching groups.

2.4 Mixture Probabilistic Matrix Factorization

In the second stage, based on the learned watching groups, we propose mPMF model to learn television's preference for TV programs to make appropriate TV recommendations. Specifically, let us first introduce our mPMF model, and then describe the inference algorithm in detail.

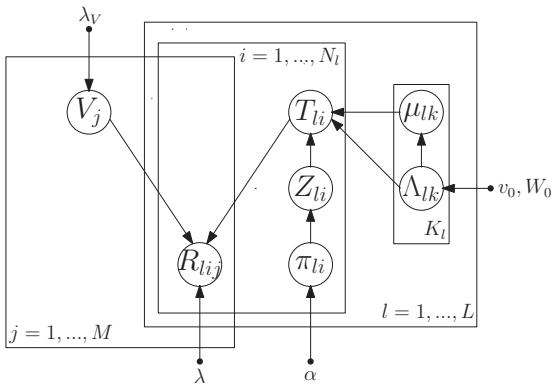


Figure 1: The Graphical Model of mPMF.

2.4.1 Model Description

In the traditional Probabilistic Matrix Factorization (PMF), both television and program latent vectors are drawn from a single Gaussian distribution if we apply it for TV recommendations. And the preference of a television for a program is modeled as the dot product of the television latent vector and the program latent vector. However, in reality, programs played at a television would be watched by one or more watching groups. The preferences of a television for TV programs could be decomposed into a mixture preference of the hidden watching groups who watch TV programs in front of this television. In other words, correctly capturing the mixture preference of underlying watching groups would help to improve the accuracy of recommendations for television. To this end, we develop the mPMF model that assumes the television-specific latent factor is drawn from a mixture of Gaussian. The mixture number for each television is the learned watching group number in the first stage. The preference of television could be represented as a weighted combination of its underlying watching groups' preference. Thus, the preference of each watching group is considered to be drawn from a different Gaussian component. Specifically, the graphical model and generative process of mPMF are shown in Figure 1 and Table 1 respectively.

To apply mPMF model, the time duration that j-th program is played at the i-th television in the l-th group would be converted to a numerical rating, denoted as R_{lij}. More mathematical notions are shown in Table 2. There are many different ways to convert time duration to numerical rating. The details of these conversion methods for this kind of implicit data would be discussed in the experimental section. And we will

Table 1: The generative process.

1. For each program j,
   a. Draw $V_j \sim \mathcal{N}(V_j \mid 0, \lambda_V^{-1}\mathbf{I})$.
2. For each group l,
   a. For each television i,
      - Draw $\pi_{li} \sim Dirichlet(\alpha)$.
      - Pick a Gaussian $Z_{li} \sim discrete(\pi_{li})$.
      - Draw $T_{li} \sim \mathcal{N}(T_{li} \mid \mu_{lz_{li}}, \Lambda_{lz_{li}}^{-1})$.
   b. For each typical user k,
      - Draw $u_{lk} \sim \mathcal{N}(u_{lk} \mid 0, (\beta_0\Lambda_{lk})^{-1})$.
      - Draw $\Lambda_{lk} \sim \mathcal{W}(\Lambda_{lk} \mid W_0, v_0)$.
3. For each non-missing entry (l, i, j),
   a. Draw $R_{lij} \sim \mathcal{N}(R_{lij} \mid T_{li}^T V_j, \lambda^{-1})$.

investigate their performances on our data set. As televisions in a group have the similar composition of watching groups, mPMF assumes their latent factors are drawn from the similar mixture of Gaussian. It means televisions in a group would share the same mean vector and covariance matrix for each Gaussian component but may have different mixing coefficients.

2.4.2 Inference We employ an EM style algorithm to infer the Maximum a posterior (MAP) estimates. Given the observation R and hyperparameters λ, λ_V, α, v_0 and W_0, the maximization of posterior is equivalent to maximizing the log-likelihood of latent parameters T, V, π, μ, Λ:

$$\begin{aligned}\mathcal{L} = &-\frac{\lambda}{2}\sum_{l,i,j} I_{lij}(R_{lij} - T_{li}^T V_j)^2 - \frac{\lambda_V}{2}\sum_j V_j^T V_j \\ &+ \sum_{l,i}\ln\sum_k \pi_{lik}\mathcal{N}(T_{li} \mid \mu_{lk}, \Lambda_{lk}^{-1}) + \sum_{l,i,k}(\alpha_k - 1)\ln\pi_{lik} \\ &+ \frac{1}{2}\sum_{l,k}\{(v_0 - D)\ln|\Lambda_{lk}| - \beta_0\mu_{lk}\Lambda_{lk}\mu_{lk}^T + Tr(W_0^{-1}\Lambda_{lk})\}.\end{aligned}$$

As the summation over k is inside the ln function for the third item in the log-likelihood, it is intractable to obtain the close-form of latent variables T_{li} and π_{lik} by maximizing the log-likelihood. To overcome this problem, we follow [17] to define $q(z_{li} = k) = \phi_{lik}$, and apply the Jensen's inequality. Thus, the third item is given by,

$$\begin{aligned}&= \ln\sum_Z p(T, Z \mid \pi, \mu, \Lambda) \\ &>= \sum_Z q(Z)\ln p(T, Z \mid \pi, \mu, \Lambda) - \sum_Z q(Z)\ln q(Z) \\ &= E_q[\ln p(T, Z \mid \pi, \mu, \Lambda)] - E_q[\ln q(Z)] \\ &= \sum_{l,i,k}\phi_{lik}\{-\frac{D}{2}\ln 2\pi + \frac{1}{2}\ln|\Lambda_{lk}| - \frac{1}{2}(T_{li} - \mu_{lk})^T\Lambda_{lk}(T_{li} - \mu_{lk}) \\ &\quad + \ln\pi_{lik} - \ln\phi_{lik}\},\end{aligned}$$

Therefore, we can relax the object function by utilizing Variational Inference method for latent variable Z as follows,

$$\begin{aligned}\mathcal{L} = &-\frac{\lambda}{2}\sum_{l,i,j} I_{lij}(R_{lij} - T_{li}^T V_j)^2 - \frac{\lambda_V}{2}\sum_j V_j^T V_j \\ &+ \sum_{l,i,k}\phi_{lik}\{-\frac{D}{2}\ln 2\pi + \frac{1}{2}\ln|\Lambda_{lk}| - \frac{1}{2}(T_{li} - \mu_{lk})^T\Lambda_{lk}(T_{li} - \mu_{lk}) \\ &+ \ln\pi_{lik} - \ln\phi_{lik}\} + (\alpha_k - 1)\ln\pi_{lik} \\ &+ \frac{1}{2}\sum_{l,k}\{(v_0 - D)\ln|\Lambda_{lk}| - \beta_0\mu_{lk}\Lambda_{lk}\mu_{lk}{}^T - Tr(W_0^{-1}\Lambda_{lk})\}\end{aligned}$$

$$s.t.\ \forall k,\ 0 \le \phi_{ik}, \pi_{lik} \le 1,\ \textstyle\sum_{k=1}^K \phi_{lik} = 1,\ \sum_{k=1}^K \pi_{lik} = 1.$$

Alternating Least Squares (ALS) is a popular optimization method, which leads to more accurate parameter estimation and faster convergence. Thus, we use ALS technique to compute each latent variable with fixing the other variables when maximizing the relaxed log-likelihood. The updating equation for each variable of interest shown in Table 3 is obtained by setting its gradient of the relaxed log likelihood to zero. The detailed algorithm for mPMF model is shown in Table 3.

Table 2: Mathematical Notations.

Symbol	Description
R_{lij}	Rating on i-th television in l-th group for j-th program
V_j	D-dimension latent factor of j-th program
T_{li}	D-dimension latent factor of i-th television in l-th group
Z_{li}	1-of-K_l binary vector, where z_{lik} indicates whether picks the k-th Gaussian component
π_{li}	K_l-dimension mixing coefficients vector
μ_{lk}	Mean of the k-th Gaussian component in l-th group
Λ_{lk}	Precision matrix of the k-th Gaussian component in l-th group
M	The number of programs
L	The number of partitioned television groups
N_l	The number of televisions in the l-th group
K_l	The number of Gaussian components in the l-th group
λ, λ_V	Parameters for Gaussian distribution
v_0, W_0	Parameters for Wishart distribution
α	Parameter for Dirichlet distribution

Step 1: Randomly initialize $\{V_j, T_{li}, \Lambda_{lk}, \mu_{lk}, \pi_{lik}, \phi_{lik}\}$.
Step 2: Execute E-Step and M-Step in each iteration repeatedly until the log-likelihood converges:

E-Step:

- $V_j = (TC_jT^T + \lambda_V\mathbf{I})^{-1}TC_jR_j$
- $T_{li} = (VC_{li}V^T + \sum_{k=1}^{K_l}\phi_{lik}\Lambda_{lk})^{-1}(VC_{li}R_{li} + \sum_{k=1}^{K_l}\phi_{lik}\Lambda_{lk}\mu_{lk})$

M-Step:

- $\pi_{lik} = \phi_{lik} + \alpha_k - 1$
  Normalize π_{lik}
- $\phi_{lik} = |\Lambda_{lk}|^{\frac{1}{2}}\pi_{lik}exp\{-\frac{1}{2}(T_{li} - \mu_{lk})^T\Lambda_{lk}(T_{li} - \mu_{lk})\}$
  Normalize ϕ_{lik}
- $\Lambda_{lk} = [W_0^{-1} + \beta_0\mu_{lk}\mu_{lk}^T + \sum_{i=1}^{N_l}\phi_{lik}(T_{li} - \mu_{lk})(T_{li} - \mu_{lk})^T]^{-1} \times [(v_0 - D + \sum_{i=1}^{N_l}\phi_{lik})\mathbf{I}]$
- $\mu_{lk} = (\beta_0 + \sum_{i=1}^{N_l}\phi_{lik})^{-1}(\sum_{i=1}^{N_l}\phi_{lik}T_{li})$

Step 3: After obtaining optimal $\hat{T}$ and $\hat{V}$, the predicted rating on i-th television of l-th group for j-th program is calculated by: $R_{lij} = \hat{T}_{li}^T\hat{V}_j$.

Table 3: The Algorithm for mPMF Model.

In the algorithm, C_{li} is a diagonal matrix with λI_{lij}, $j = 1..., M$ as its diagonal elements, I_{lij} is the indicator function that is equal to 1 if the i-th television in the l-th group rated the j-th program and equal to 0 otherwise, and $R_{li} = (R_{lij})_{j=1}^M$. Specifically, the basic framework of proposed algorithm for mPMF model is as follows: given the partitioned television groups, watching group number in each television group and the converted

numerical ratings as input, we first initialize the latent variables randomly, then employ EM style algorithm by updating each variable of interest in each iteration repeatedly until the relaxed log-likelihood converges. After obtaining the estimated television-specific and program-specific latent factor $\hat{T}$ and $\hat{V}$, we can make predictions on those unplayed TV programs, and then provide recommendations to televisions.

3 Experimental Results

In this section, we evaluate the performance of mPMF model with a real-world data set.

3.1 Data Preprocessing

In this paper, we use one-week TV data (i.e., from 03/12 to 03/16 in 2014) to evaluate the performance of the proposed mPMF model, which is collected from our industry partner. We do the following preprocessing on our data set: 1) removing watching records with the watching duration time less than 11 seconds; 2) removing televisions that have played less than 21 programs or the televisions whose playing activity span is less than 4 days; 3) removing programs that are played by less than 1,001 televisions. After the preprocessing, we have 230,196 televisions, 4,289 different TV programs and 14,159,678 playing records. Totally, these TV programs consist of 8 first-level genres and 211 sub-genres. In addition, we randomly select 10% of playing records as testing and train our model with the remaining.

3.2 Evaluation Metrics

The model is evaluated in terms of prediction accuracy, ranking accuracy and Top-K recommendation performance.

Prediction Accuracy. RMSE measures the prediction accuracy. $RMSE = \sqrt{\frac{\sum_{l,i,j}(R_{lij}-\hat{R}_{lij})^2}{N}}$, where R_{lij} and $\hat{R}_{lij}$ denote the observed and predicted rating, and N is the number of test data.

Ranking Accuracy. Kendall Tau Coefficients (Tau), Normalized Discounted Cumulative Gain (nDCG), and Degree of Agreement (DOA) [9] are used to measure the overall ranking accuracy. Suppose each estate i is associated with a groundtruth rating x_i and a predicted rating y_i. There are n pairs. Each pair $< i, j >$ is said to be concordant, if both $x_i > x_j$ and $y_i > y_j$ or if both $x_i < x_j$ and $y_i < y_j$. Also, it is said to be discordant, if both $x_i > x_j$ and $y_i < y_j$ or if both $x_i < x_j$ and $y_i > y_j$. Tau is defined as $\frac{\#conc-\#disc}{\frac{1}{2}n(n-1)}$. DCG is defined as $\sum_{i=1}^{N}\frac{2^{rel_i}-1}{log_2(1+i)}$, where rel_i is the relevant score and regarded as the testing rating in our experiments. Given the ideal DCG (IDCG), DCG of groundtruth ratings, nDCG is defined as $\frac{DCG}{IDCG}$.

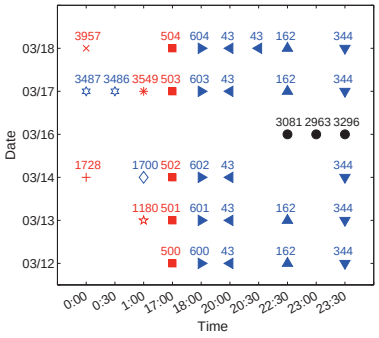
Top-K Recommendation. We will use cumulative distribution (CD) [8], Precision@K, and Recall@K to evaluate the Top-K recommendation performance. MAP, the mean of average precision(AP) over all televisions in the testing, is also used here. CD is computed as follows: 1) $\mathcal{M}$ top highest ratings in the testing are selected. 2) For each program j with the selected rating by television t, we randomly select $\mathcal{C}$ additional programs and predict the ratings by television t for j and other $\mathcal{C}$ programs. 3) We sort these programs based on their predicted ratings in decreasing order. There are $\mathcal{C}+1$ different possible ranks for program j, ranging from the best case 0% to the worst case 100%. Thus we can get the cumulative distribution associated with the rank. In our experiments, we specify $\mathcal{M} = 100,000$ and $\mathcal{C} = 2,000$. Moreover, AP, Precision@K and Recall@K are calculated as $AP_t = \frac{\sum_{i=1}^{N} p(i)\times rel(i)}{\#relavantprograms}$, $Precision@K = \frac{\sum_{T_t\in T}|T_K(T_t)|}{\sum_{T_t\in T}|R_K(T_t)|}$, $Recall@K = \frac{\sum_{T_t\in T}|T_K(T_t)|}{\#relavantprograms}$ respectively, where $R_K(T_t)$ are the top-K programs recommended to television t, $T_K(T_t)$ denotes all truly relevant programs among $R_K(T_t)$, T represents the set of televisions in the testing, i is the position in the rank list, N is the number of returned items in the list, $p(i)$ is the precision of a cut-off rank list from 1 to i, and $rel(i)$ is an indicator function that equals to 1 if the program is relevant, otherwise equals to 0. The term *relevant programs* in our experiments is defined as the programs in the testing.

3.3 Baseline Methods

To demonstrate the effectiveness of our mPMF model, we construct three baseline methods based on mPMF model. First, we randomly divide televisions into L groups, and assume the number of watching groups in each television group is either fixed or random. Thus we construct the first two baseline methods where the number of watching groups in each television group is fixed to 1 and 3, denoted as mPMF@1 and mPMF@3, respectively. The third baseline method is considering the number of watching groups as a random value, ranging from 1 to 10, which is denoted as mPMF@random. In addition, we also adopt PMF [15] as another baseline method. Specifically, PMF in our experiments learns the television and program latent vectors which are both drawn from a single Gaussian distribution. Totally, we have four baseline methods for evaluating the recommendation performance of mPMF model.

3.4 Discovery of Watching Groups

In this section, we show an example of the learned watching group. After performing clustering on the training data set, we obtain about 12,000 television



ID	Name
500-504	The Chew
600-604	General Hospital
43	Jeopardy!
162	ABC World News With Diane Sawyer
344	Wheel of Fortune
3081	House Hunters International
2936	House Hunters International
3296	House Hunters

ID	Genre
500-504	Talk-Show
600-604	Drama
43	Game-Show
162	News
344	Game-Show
3081	Reality-TV
2936	Reality-TV
3296	Documentary

Figure 2: An example of clustering: Left is the clustering result, and Right is the corresponding program names and main genres.

groups, for each of which we learn a number of hidden watching groups. We randomly select the clustering result of a television and show in figure 2.

In figure 2, the left is the clustering result. The x-axis is the time for program to start playing, and y-axis is date where 03/16 is Sunday and others are the week days. We use each marker to represent a program that has been watched and the number above it is the program ID. Each color of makers indicates a different watching group (cluster), and each shape of markers is a different TV program (TV shows with different episodes have the same kind of shape). Due to the limited space, we only list their corresponding program names and main genres for some corresponding program IDs in the right table. From the result, we observe that there are 3 groups of audiences who watch programs in front of the television: Group 1 (red) would like to watch talk show **The Chew** related to cooking every workday around 17:00; Group 2 (blue) prefers to watch game show, such as **Jeopardy!** and **Wheek of Fortune** at about 20:00 and 23:30 respectively in each workday. This watching group also likes to watch drama and news every night in workday; Group 3 (black) watches **House Hunters** on Sunday night. Based on the observation, we can guess this television is supposed to be placed in a household which has a couple. The wife in the household likes watching TV show regarding cooking. The husband prefers news, drama and game show. They might watch TV on weekend together. As can be seen from this example, television's preference for TV programs actually results from the preference of its underlying watching groups who watch TV programs in front of this television. And we may identify these watching groups via clustering TV play records.

3.5 Performance Comparisons

To apply mPMF model, we first convert the watching time duration for each program into a numerical rating as follows. We calculate the ratio of the actual watching time to the total time of a program in each record. For example, if the overall time of a program is 60 mins and it is watched at one TV for 30 mins, the ratio is calculated as 0.5. And we use the computed ratio as a rating. Moreover, if a single program is watched more than once at a TV, the sum over its multiple ratios is regarded as the rating. For instance, if one 60-min program is watched at a TV twice (10 mins and 20 mins respectively), the converted rating is 3/6 (i.e., 1/6 + 2/6). After this data transformation, we get 12,351,042 ratings, among which 1,214,202 ratings are used as testing in the experiments. Based on this conversion method, we will evaluate our model with other baseline methods on precision accuracy, ranking accuracy and Top-K recommendation performance.

3.5.1 Prediction Accuracy Performance

We examine the prediction accuracy performance in different models. Figure 3 shows the comparisons of RMSE for our model verse baseline methods with latent factor dimensions ranging from 10 to 100. The lower RMSE indicates the better prediction to the groundtruth ratings. As most observed ratings are less than 1, the RMSE values totally are quite low. From the result, we can see the mPFM model achieves the best performance, while the PMF model performs worst. For example, when the latent factor dimension is 50 and 60, mPMF is increased by 9.65% and 9.83%, respectively. The main reason why mPMF could make more accurate predictions than PMF is that mPMF models the television latent features by assuming the preference of television is the mixture preference of its hidden watching groups. Also, mPMF@1 method gets a better performance than PMF, but worse than others. Although mPMF@1 method takes the television group information into account in the model training, the assumption that the television latent feature in each group is drawn from a shared Gaussian distribution somehow is similar to the one of PMF model. This is the reason why mPMF@1 causes worse performance than others. In addition, we observe that both mPMF@3 and mPMF@random obtain the similar performance. And they are worse than mPMF model, but better than mPMF@1 and PMF models. Both of them are dividing the preference of a television into several watching groups' preference, but they could not capture the actual watching groups for each television, resulting in worse performance than mPMF.

3.5.2 Ranking Accuracy Performance

Figure 4 and Figure 5 show the overall ranking performance of mPMF and other four baseline methods in terms of Tau and nDCG, respectively. The higher value

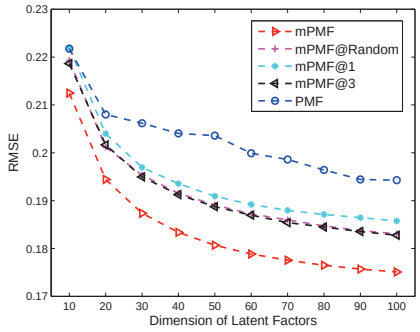


Figure 3: RMSE

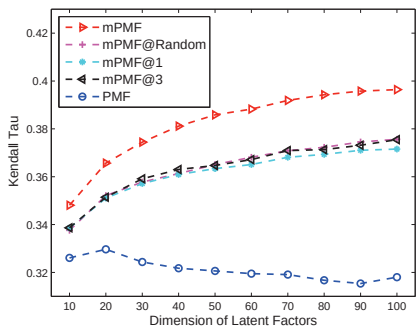


Figure 4: Kendall Tau

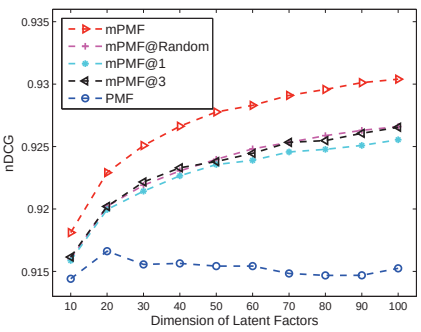


Figure 5: nDCG

on both metrics is desired. We can observe that mPMF outperforms the other baselines on both two metrics and PMF has the worst performance. Particularly, with latent factor dimension increasing, the performance of mPMF, mPMF@1, mPMF@3 and mPMF@random based on Tau and nDCG improves rapidly; however, PMF performs worse and worse. Thus, the difference of performance between mPMF-based methods and PMF becomes larger and larger when latent factor dimension increases. Specifically, when latent factor dimension is 100, mPMF achieves 39.64% in terms of Tau, while PMF is only 31.80%. It demonstrates that mining the underlying watching groups of a television and then decomposing television's preference into the preference of its corresponding watching groups, would help recommendation system to make more accurate TV recommendations. Based on the result, we also can see that mPMF@3 and mPMF@random have similar performance, and their performance is slightly better than mPMF@1. Thus, we can obtain the following conclusion: 1) the preference of a television is the combination of the preference of corresponding watching groups; and 2) mining actual watching groups for a television could improve the recommendation's accuracy.

3.5.3 Top-K Recommendation Performance

CD is used to evaluate the quality of top-K recommendations for different models. Recommendation system only recommends limited TV programs for a television, so we only focus on CD performance within top-6% and report the result in Figure 6 and 7 with 10D and 30D latent factors, respectively. When the latent factor dimension increases, the performance of different models become much better. From the result, mPMF gets the best performance, but PMF performs worst. For example, when the latent factor dimension is 10 and the rank is top-1%, the CD of mPMF is 61.08% while the CD of PMF is only 54.48%. Moreover, it shows that mPMF@3 and mPMF@random perform similarly, and both of them outperform mPMF@1. The CD performance of different methods keep consistent with before. It further illustrates that the involvement of watching groups would help to learn more accurate televisions' preference.

Table 4 shows the comparisons of all methods in terms of Precision@K, Recall@K, and MAP with 10D, 30D and 60D latent features. Performances in terms of Precision@K, Recall@K are evaluated with different K values, that is, $K = 5$ and $K = 10$. Totally, mPMF performs the best among all methods on these three metrics. Based on the result of MAP, the performance of five methods is similar to the result performed on other metrics shown before. When the K increases, the precision decreases while the recall increases. For precision and recall, mPMF@random performs nearly as well as mPMF. As the watching group number of the television in mPMF@random method might be larger than the number that has been learned in the first step, it causes the television latent factor might be drawn from a mixture associated with more Gaussian distributions. In this case, mPMF@random might achieve the same effect as mPMF when the mixing coefficients on those extra Gaussian components are approximate to zero.

3.6 Performance Comparisons with Different Data Conversion Methods

In this section, we will evaluate different methods for converting a observed time slot that a TV program is watched into a numerical rating. In addition to the conversation method used in above section, denoted as *Ratio*, there are other three kinds of conversion methods. The second method is using frequency as a numerical rating which refers to how many times a program is watched, denoted as *Frequency*. The rating in the third method is a simple boolean value, which is equal to 1 if the program is watched and equal to 0 otherwise. This method is denoted as *Binary*. Another method is proposed by [5], treating the implicit data as the indication of positive and negative preference regarding different confidence levels. We use the value retrieved by the first method as the confidence level. We denote this kind of method as *Confidence*. The scales of ratings on four methods are different and the observed ratings in *Binary* method are always 1. Therefore, the performance comparisons in terms of

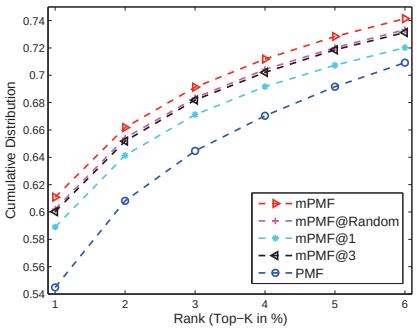


Figure 6: CD (D = 10)

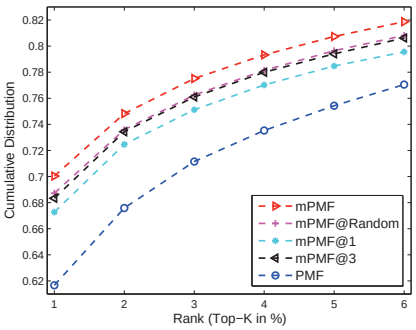


Figure 7: CD (D = 30)

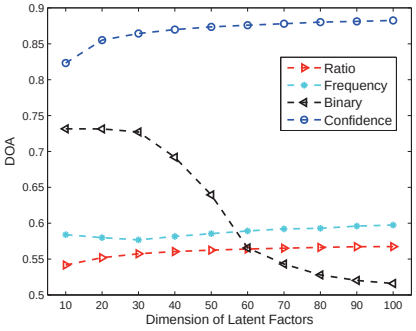


Figure 8: DOA on mPMF

	Precision@5	Recall@5	Precision@10	Recall@10	MAP
10D Latent Features					
mPMF	**0.0432**	**0.0411**	0.0277	0.0527	**0.0377**
mPMF@Random	0.0430	**0.0411**	**0.0284**	**0.0540**	0.03619
mPMF@1	0.04105	0.0391	0.0259	0.0494	0.0342
mPMF@3	0.0429	0.0409	0.0281	0.0534	0.0354
PMF	0.0320	0.0304	0.0211	0.0402	0.0276
30D Latent Features					
mPMF	0.0517	0.0493	0.0322	0.0613	**0.0469**
mPMF@Random	**0.0529**	**0.0503**	**0.0326**	**0.0620**	0.0452
mPMF@1	0.0488	0.0464	0.0296	0.0564	0.0417
mPMF@3	0.0516	0.0491	0.0318	0.0606	0.0439
PMF	0.0392	0.0373	0.0242	0.0461	0.0359
60D Latent Features					
mPMF	**0.0584**	**0.0556**	**0.0356**	**0.0679**	**0.0534**
mPMF@Random	0.0581	0.0553	0.0352	0.0671	0.0497
mPMF@1	0.0534	0.0508	0.0316	0.0603	0.0457
mPMF@3	0.0562	0.0535	0.0338	0.0643	0.0479
PMF	0.0499	0.0475	0.0292	0.0556	0.0466

Table 4: Performance Comparisons(Precision, Recall, MAP).

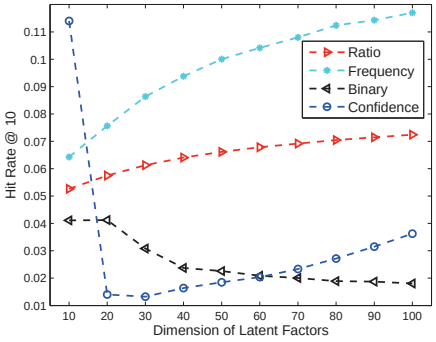


Figure 9: Hit Rate on mPMF.

RMSE, nDCG, Tau become meaningless. Then we will examine the performance of four conversion methods on mPMF model in terms of DOA and Hit Rate. $Hitrate@K$ (Recall@K) in fact indicates the recall value of the recommended top-K of TV programs. Here, we set $K = 10$. The performance comparisons for different methods based on DOA and Hit Rate with different latent factor dimensions are shown in Figure 8 and Figure 9, respectively. The higher values on both metrics indicate better performance.

From the result, we can see that $Confidence$ performs much better than other methods in terms of DOA, but it gets the worst hit rate performance in low latent factor dimensions, not including 10D. The optimization in $Confidence$ method spends varying granularity on minimizing the distance to the observed ratings, and also accounts for the unobserved data, which results in good ranking performance. Although the confidence level is used during the optimizing process, it simply employs binary value to represent the television's preference. Thus, its top 10 recommendation performance is possibly not good as DOA performance. Also, it is interesting to find that the performance of *Binary* becomes worse and worse when the latent factor increases. The main reason is that simply using binary to express the preference for a program usually ignores a lot of other useful information, especially when more latent factors are taken into account. In addition, we can observe that the performance of *Ratio* and $Frequency$ keeps consistent on both DOA and Hit Rate. *Ratio* is more reasonable because it includes the meaningful information, i.e. how long a program is watched. But its most of ratings are between 0 and 1, so it is hard to distinguish the degree on the preference for a program, in particular for the movie that usually is played only one time in TV.

4 Related Work

Generally, the related work of this paper can be grouped into two categories. The first category is about collaborative filtering (CF) models. Indeed, CF is widely used in recommendation systems, which mainly includes memory-based and model-based approaches. In the memory-based approaches, some neighbouring ratings are used to generate the rating prediction. The model-based approaches predict ratings based on the matrix factorization models learned from training data. Indeed, many matrix factorization methods have been proposed for CF, such as low rank approximation [6, 18], and latent factor based methods [8, 15]. Specifically, the later is using the learned user and item latent vectors to predict the unknown ratings. For example, PMF [15] is proposed by assuming the observed ratings are drawn from a Gaussian distribution, and each user and item is modeled as a latent factor with Gaussian prior. As

MAP easily leads to over-fitting, some other inference approaches are developed to learn more accurate latent user and item factors, such as [14, 3]. Similar to the movie recommendation, PMF also could be used for TV recommendation, where each TV program is regarded as an item and each television is considered as a user. However, PMF simply regards each television as a single individual user, which neglects the involvement of watching groups who are watching TV programs in front of the television.

The second category is about TV program recommendations. Instead of using explicit ratings, TV recommendation system would use implicit feedback reflecting opinion from the observed behaviour [11], such as the watching time slot, to learn user's preference. Recently, there are many studies about TV recommendations having been reported. In addition to the methods based on time series [4], some other algorithms based on PMF are proposed to handle implicit data, such as [5, 19]. For example, Hu *et al* [5] proposed to transform the implicit user behaviour into user's preference and confidence level so that both observed and unobserved user behaviours could be accounted for the model. It assumes the confidence level is the indication degree that the user indeed likes the TV program. Furthermore, Xin *et al* [19] proposed a PMF based model to learn the user-program matrix, assuming there may be multiple ratings for a same program, such as a TV show with different episodes. However, these algorithms concern more about how to handle implicit feedback for TV data. Thus, they also can be applied in our proposed model to improve the recommendation accuracy.

5 Conclusion

In this paper, we developed a novel two-stage framework for building personalized TV recommender system. Specifically, we first proposed to automatically learn the number of latent watching groups for televisions by clustering watching records. Then, the mPMF model was proposed to learn the mixture preference of television for programs, where television latent vector is assumed to be drawn from a mixture of Gaussian and the mixture number is the number of learned watching groups. Particularly, televisions in a group share the same group of Gaussian components but have different mixture weights. Finally, experimental results based on a real-world TV data set clearly demonstrate the effectiveness and robustness of our model.

6 Acknowledgements

This research was supported in part by National Institutes of Health under Grant 1R21AA023975-01 and National Natural Science Foundation of China under Grant 61203034.

References

[1] W. Chen, Y. Zhang, and H. Zha. Mining iptv user behaviors with a coupled lda model. In *IEEE*, 2013.

[2] G. Hamerly. Making k-means even faster. In *SIAM International Conference on Data Mining (SDM)*, 2010.

[3] S. Hanhuai and B. Arindam. Generalized probabilistic matrix factorizations for collaborative filtering. In *ICDM*, pages 1025–1030, 2010.

[4] L. Helmut. *New introduction to multiple time series analysis*. Springer, 2007.

[5] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, pages 263–272, 2008.

[6] R. H. Keshavan, A. Montanari, and S. Oh. Matrix completion from noisy entries. *CoRR*, 2009.

[7] E. Kim, S. Pyo, E. Park, and M. Kim. An automatic recommendation scheme of tv program contents for (ip)tv personalization. In *IEEE*, 2011.

[8] Y. Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *SIGKDD*, pages 426–434, 2008.

[9] Q. Liu, E. Chen, H. Xiong, D. C.H.Q., and J. Chen. Enhancing collaborative filtering by user interest expansion via personalized ranking. In *IEEE*, 2012.

[10] D. Luo, H. Xu, H. Zha, J. Du, R. Xie, X. Yang, and W. Zhang. You are what you watch and when you watch: Inferring household structures from iptv viewing data. In *IEEE*, pages 61–72, 2014.

[11] D. Oard and J. Kim. Implicit feedback for recommender systems. In *AAAI*, pages 81–83, 1989.

[12] S. Pyo, E. Kim, and M. Kim. Automatic recommendation of (ip)tv program schedules using sequential pattern mining. In *Adjunct Proceedings of EuroITV*, 2009.

[13] S. Pyo, E. Kim, and M. Kim. Lda-based unified topic modeling for similar tv user grouping and tv program recommendation. In *IEEE*, 2014.

[14] S. Ruslan and M. Andriy. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *ICML*, pages 880–887, 2008.

[15] S. Ruslan and M. Andriy. Probabilistic matrix factorization. In *NIPS*, 2008.

[16] S. van Dongen. *Graph Clustering by Flow Simulation*. PhD thesis, University of Utrecht, May 2010.

[17] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *KDD*, 2011.

[18] L. Wu, E. Chen, Q. Liu, L. Xu, T. Bao, and L. Zhang. Leveraging tagging for neighborhood-aware probabilistic matrix factorization. In *CIKM*, pages 1854–1858. ACM, 2012.

[19] Y. Xin and H. Steck. Multi-value probabilistic matrix factorization for ip-tv recommendations. In *RecSys*, 2011.

[20] G. Yu, T. Westholm, M. Kihl, I. Sedano, A. Aurelius, C. Lagerstedt, and P. Odling. Analysis and characterization of iptv user behavior. In *IEEE*, 2009.