Temporal Probabilistic Matrix Factorization

Robert Timpe
rtimpe@cs.ucsd.edu

Kristjan Jonsson
kjonsson@eng.ucsd.edu

Joseph Geumlek
jgeumlek@eng.ucsd.edu

Mark Qiao
faqiao@eng.ucsd.edu

Abstract

Matrix Factorization (MF) is a common method used in recommendation systems. Given rating data where users each rate a subset of items MF models try to learn latent features of both users and items that can be used to predict ratings for unobserved (user, item) pairs.

In this project, we explore two extensions to the basic Probabilistic Matrix Factorization (PMF) model. First, we describe a mixture model where users are assigned to "profiles" which represent groups of users. This accounts for the fact that groups of users often have very similar rating habits. The second model extends the mixture model by giving the profile assignment Markov dynamics over time. This accounts for the fact that user preferences may change over time.

1. Introduction

Matrix factorization approaches have been successfully applied to recommendation problems, including the famous Netflix Prize [3]. We focus on Probabilistic Matrix Factorization (PMF) [4] and its applications to the recommendation problem.

We start be describing the original Probabilistic Matrix Factorization model from Salakhutdinov-et.al [4].

1.1. Probabilistic Matrix Factorization

Figure 1 shows the Probabilistic Matrix Factorization (PMF) presented by Salakhutdinov et.al. Let M be the number of movies and N the number of users. Let $R_{ij} \in \{1, 2, 3, 4, 5\}$ be the rating user i gave to movie j. Let $U \in \mathcal{R}^{N\times k}$ be the latent user features and $V \in \mathcal{R}^{M\times k}$ be the latent movie features with U_i and V_j as the latent features for user i and movie j respectively in column format.

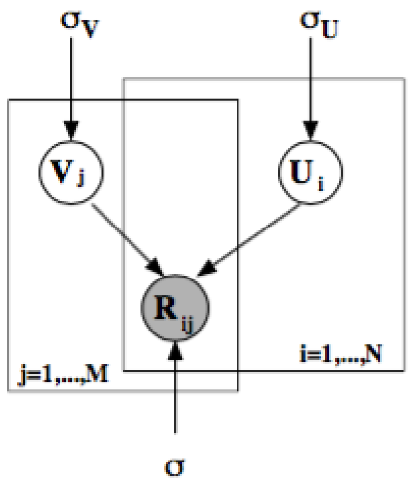

Figure 1: Graphical model for PMF from the original paper.

The likelihood of observed ratings is

$$p(R|U, V, \sigma^2) = \prod_{i=1}^{N}\prod_{j=1}^{M} \left[\mathcal{N}(R_{ij}|U_i^T V_j, \sigma^2)\right]^{I_{ij}}$$

where $\mathcal{N}(x|\mu, \sigma^2)$ is the normal distribution and I_{ij} is the indicator function that is equal to 1 if user i rated movie j and equal to 0 otherwise. We assume a Gaussian prior over user and item vectors, namely

$$p(U|\sigma_U^2) = \prod_{i=1}^{N} \mathcal{N}(U_i|0, \sigma_U^2\mathbf{I}),$$

and

$$p(V|\sigma_V^2) = \prod_{j=1}^{M} \mathcal{N}(V_j|0, \sigma_V^2\mathbf{I}).$$

It can be shown that finding the MAP is equivalent to minimizing the following regularized squared objective

$$l(U,V) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2$$
$$+ \frac{\lambda_U}{2}\sum_{i=1}^{N}\|U_i\|_2^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}\|V_j\|_2^2,$$

where $\|\cdot\|_2^2$ is the Frobenius Norm.

1.2. Motivation

We propose two extensions to the PMF model. We first consider a mixture model in which latent representations for users are replaced with latent representations for user clusters which we call profiles, with every user being assigned to a single profile. We hope that this model will provide a sort of regularization by reducing the number of latent representations to learn. Additionally, this model may help with the cold start problem by using data from users with many ratings to help predict users with few ratings. We derive an EM algorithm for learning parameters and hidden features in the model. We also experimented with a fully Bayesian version of this model with added hyper-priors that was trained using Gibbs sampling.

The second model we describe is an HMM that allows the user profile assignments to evolve over time. We hope that this model will be able to capture changes in user preferences over time. This model will also be able to more easily account for situations where a user watches and rates an uncharacteristic set of movies in a short amount of time. We perform inference in this model using a Gibbs sampler.

2. Extensions to PMF

We now describe our extensions in more detail.

2.1. Mixture PMF

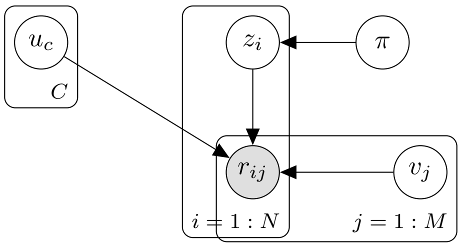

Figure 2: Graphical model for the mixture PMF. Hyperparameters are omitted for clarity

We give three motivations for a mixture model extension to the PMF.

- With a large number of users we expect some of them to have similar preferences. For example, a few users might predominantly like superhero movies. Instead of having a separate latent vector for each user, we have users share latent representations to compress the model and prevent the model from exploding in size with an increasing number of users. This may also help with generalization error.
- We might be interested in clustering users for analysis.
- When a new user enters the system after his first rating we can use the posterior-predictive distribution to average our predictions over the different mixture components to get a reasonable rating. This helps with the cold start problem.

Figure 2 shows the generative model. $z_i \in \{1,\ldots,C\}$ is the hidden mixture state of user i with,

$$P(z_i|\pi) = \text{Discrete}(\pi), \quad \pi \in S(C),$$

where π has prior

$$P(\pi|\alpha) = \text{Dirichlet}(\alpha), \quad \alpha \in \mathcal{R}_+^C.$$

Under this model the likelihood of the ratings becomes:

$$P(r_{ij}|U, v_j, z_i) = \prod_{c=1}^{C}\left[\mathcal{N}(r_{ij}|u_c^T v_j, 1)\right]^{I(z_i,c)}$$

where $I(z_i,c)$ is an indicator function on whether user i is assigned to cluster c. We assume a zero-centered Gaussian prior on the latent vectors with precision $\lambda \in \mathcal{R}_+$. Using EM we can estimate the MAP for the parameters U, V, π and the posterior for the hidden states z_i. For the E-step we need to calculate the posterior over z_i which we denote $\tilde{p}_c^{(i)}$, i.e.

$$\tilde{p}_c^{(i)} = P(z_i = c|U,V,\pi,R) \propto \pi_c \prod_{j=1}^{M}\left[\mathcal{N}(r_{ij}|u_c^T v_j, 1)\right]^{I_{ij}}.$$

The unnormalized log-posterior is

$$\ln\left(\tilde{p}_c^{(i)}\right) = \ln(\pi_c) - \frac{1}{2}\sum_{j=1}^{M} I_{ij}(r_{ij} - u_c^T v_j)^2.$$

In the M-step we minimize the expected negative log-posterior

$$l(U,V) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}\sum_{c=1}^{C}\tilde{p}_c^{(i)}(r_{ij} - u_p^T v_j)^2$$
$$+\frac{\lambda}{2}\|U\|_2^2 + \frac{\lambda}{2}\|V\|_2^2,$$

which is a weighted least squares problem. The posterior for π is a Dirichlet because of the conjugate prior and we get the MAP estimate:

$$\pi_c = \frac{\sum_{i=1}^{N}\tilde{p}_c^{(i)} + \alpha_c - 1}{N + \sum_{c=1}^{C}\alpha_c - C}.$$

2.2. Bayesian Mixture Model

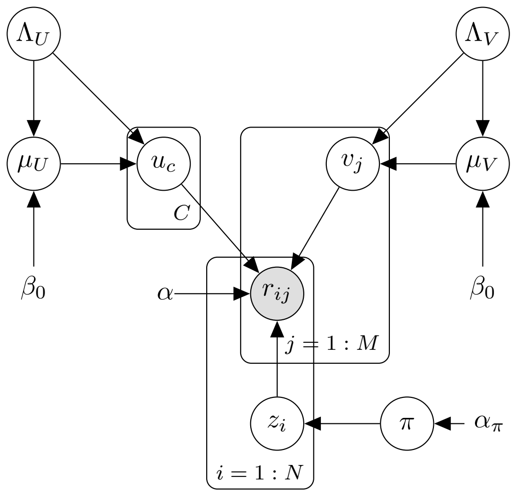

Figure 3: Graphical model for mixture BPMF

In the previous section we used the EM algorithm to get a MAP estimate of the parameters and a posterior over mixture components. The weakness of this approach is that it doesn't take in to account the uncertainty of the parameters. Using Gibbs sampling we can approximate the full posterior over the parameters and make predictions using the posterior-predictive. For our previous mixture model we hand tuned hyper-parameters to achieve the best RMSE. By adding hyper-priors to the model we can avoid having to train multiple models for different hyper-parameter settings since the posterior-predictive distribution will average over likely values of the priors.

Figure 3 shows this graphical model. It's a version of the Bayesian Probabilistic Matrix Factorization (BPMF) model described by Salakhutdinov et. al. [5] but adapted to our mixture model extension.

Under this model the priors for the latent vectors become

$$
\begin{aligned}
P(u_c|\mu_U, \Lambda_U) &= \mathcal{N}(u_c|\mu_U, \Lambda_U) \\
P(v_j|\mu_V, \Lambda_V) &= \mathcal{N}(v_j|\mu_V, \Lambda_V)
\end{aligned}
$$

where $\mu \in \mathcal{R}^k$ and $\Lambda \in \mathcal{R}^{k \times k}$ have hyperpriors

$$
\begin{aligned}
P(\mu|\Lambda) &= \mathcal{N}(\mu|0, (\beta_0 \Lambda)^{-1}) \\
P(\Lambda) &= \mathcal{W}(\Lambda|I, k)
\end{aligned}
$$

where $\beta_0 \in \mathcal{R}_+$ is a hyperparameter and $\mathcal{W}$ the Wishart distribution:

$$
\mathcal{W}(\Lambda|W, \nu) = \frac{1}{C} |\Lambda|^{(\nu-k-1)/2} \exp\left(-\frac{1}{2}\text{Tr}(W^{-1}\Lambda)\right)
$$

2.2.1 Gibbs sampling

We use Gibbs sampling to estimate the posteriors for the parameters and hidden mixture states. Gibbs sampling works by sampling from the posterior of each variable one at a time.

Salakhutdinov et al. [5] derive the posterior for the latent vectors and priors for the BPMF model which all have Gaussian distributions. Following their derivation we adapt the posterior to our mixture extension which becomes:

$$
P(v_j|R_{:,j}, U, z, \alpha) = \mathcal{N}\left(v_j|\mu_j^*, [\Lambda_j^*]^{-1}\right),
$$

where

$$
\begin{aligned}
\Lambda_j^* &= \Lambda_V + \alpha \sum_{i=1}^{N} \sum_{c=1}^{C} I_{ij} I(z_i, c) u_c u_c^T \\
\mu_j^* &= [\Lambda_j^*]^{-1} \left(\alpha \sum_{i=1}^{N} \sum_{c=1}^{C} I_{ij} I(z_i, c) u_c r_{ij} + \Lambda_V \mu_V\right).
\end{aligned}
$$

Likewise

$$
P(u_c|R_{c,:}, V, z, \alpha) = \mathcal{N}\left(u_c|\mu_c^*, [\Lambda_c^*]^{-1}\right),
$$

where

$$
\begin{aligned}
\Lambda_c^* &= \Lambda_U + \alpha \sum_{i=1}^{N} \sum_{j=1}^{M} I_{ij} I(z_i, c) v_j v_j^T \\
\mu_c^* &= [\Lambda_c^*]^{-1} \left(\alpha \sum_{i=1}^{N} \sum_{j=1}^{M} I_{ij} I(z_i, c) v_j r_{ij} + \Lambda_U \mu_U\right).
\end{aligned}
$$

The posteriors for μ_V and Λ_V are

$$
\begin{aligned}
P(\mu_V|V, \Lambda_V) &= \mathcal{N}\left(\mu_V|\mu^*, [\beta^* \Lambda_V]^{-1}\right) \\
P(\Lambda_V|V, \mu_V) &= \mathcal{W}(\Lambda_V|W^*, \nu^*)
\end{aligned}
$$

where

$$
\begin{aligned}
&\nu^* = k + M, \quad \beta^* = \beta_0 + M, \quad \mu^* = \frac{M}{\beta_0 + M}\bar{v} \\
&[W^*]^{-1} = I + M\bar{S} + \frac{\beta_0 M}{\beta_0 + M}\bar{v}\bar{v}^T \\
&\bar{v} = \frac{1}{M}\sum_{j=1}^{M} v_j, \quad \bar{S} = \frac{1}{M}\sum_{j=1}^{M}(v_j - \bar{v})(v_j - \bar{v})^T.
\end{aligned}
$$

The posteriors for μ_U and Λ_U are analogous.

The posterior for z_i can be found with

$$
P(z_i = c|U, V, \pi, R) \propto \pi_c \prod_{j=1}^{M} \left[\mathcal{N}(r_{ij}|u_c^T v_j, \alpha)\right]^{I_{ij}}
$$

The posterior for π is

$$
P(\pi|\{z_i\}, \alpha_\pi) = \text{Dirichlet}(C_0 + \alpha_\pi),
$$

Where C_0 is the count vector of cluster assignments.

2.3. Temporal model

User preferences can change over time. It's tempting to discretize time into T separate bins and define $U^{(t)} \in \mathcal{R}^{N\times k}$ to be the user factors at time t. However, the number of user parameters then becomes NkT i.e. they grow linearly with the number of time-steps, T. The model size quickly become intractable in this setting. Additionally, user preferences might be similar across time as well as between users. To account for this we extend our Bayesian mixture model by adding HMM dynamics over the mixture component $z_i^{(t)} \in \{1,\dots,C\}$. Under this scheme the number of mixture parameters $U \in \mathcal{R}^{C\times k}$ is fixed at Ck.

The mixture component transitions are described by:

$$P(z_i^{(t)}|z_i^{(t-1)}=c) = \text{Discrete}(a_c), \quad t>0,$$
$$P(z_i^{(0)}) = \text{Discrete}(\pi)$$

where the parameters $\pi, a_i \in S(C)$ have priors

$$P(\pi|\alpha_\pi) = \text{Dirichlet}(\alpha_\pi),$$
$$P(a_c|\alpha_c) = \text{Dirichlet}(\alpha_c)$$

with hyperparameters $\alpha_\pi, \alpha_c \in \mathcal{R}_+^C$.

2.3.1 Gibbs sampling

We use Gibbs sampling to approximate the posterior over the latent variables.

The posteriors are for the HMM parameters are

$$P(\pi|\{z_i^{(0)}\}) = \text{Dirichlet}(\pi|C_0+\alpha_\pi),$$
$$P(a_c|\{z_i^{(t)}\}) = \text{Dirichlet}(\pi|C_c+\alpha_c),$$

where $C_0 \in \mathcal{N}^C$ are the initial state counts i.e. $C_0(c) = \sum_{i=1}^N I(z_i^{(0)}, c)$ and $C_c \in \mathcal{N}^C$ are the transition counts from state c, i.e. $C_c(d) = \sum_{i=1}^N \sum_{t=1}^T I(z_i^{(t-1)}, c) I(z_i^{(t)}, d)$ for user i.

The posteriors for the movie and profile latent vectors are not significantly different from those derived in the previous section. The only difference is that in addition to summing over clusters and users (for the movie latent vectors) or users and movies (for the profile latent vectors) we also sum over the T time steps, taking contributions from the movies that were rated at each time step. For brevity, we do not reproduce the equations here.

For sampling the hidden mixture components $z_i^{(t)}$ we use the Forwards-Filtering Backwards-Sampling algorithm to jointly sample from $P(z_i^{(0)},\dots,z_i^{(T)}|R^{(0)},\dots,R^{(T)})$ as described by [5].

For the Forwards-Filtering pass we compute $\alpha_{t,c} = P(z_i^{(t)}=c|R^{(1)},\dots R^{(t)})$ using the recursive formula

$$\alpha_{t,d} \propto P(R^{(t)}|z_i^{(t)}=d)\sum_{c=1}^{P}\alpha_{t-1,c}a_{c,d}$$
$$= \left(\prod_{j=1}^{M}\left[\mathcal{N}(r_{ij}^{(t)}|u_d^T v_j,\alpha)\right]^{I_{ij}}\right)\sum_{c=1}^{C}\alpha_{t-1,c}a_{c,d}$$

and base case

$$\alpha_{0,c} \propto \pi_c \prod_{j=1}^{M}\left[\mathcal{N}(r_{ij}^{(0)}|u_c^T v_j,1)\right]^{I_{ij}}.$$

In log space the unnormalized probabilities become

$$\ln(\alpha_{0,c}) = \ln(\pi_c) - \frac{1}{2}\sum_{j=1}^{M} I_{ij}(r_{ij}^{(0)} - u_c^T v_j)^2,$$

$$\ln(\alpha_{t,d}) = -\frac{1}{2}\sum_{j=1}^{M} I_{ij}(r_{ij}^{(t)} - u_d^T v_j)^2 + \ln\left(\sum_{c=1}^{C}\alpha_{(t-1),c}a_{c,d}\right)$$

Once we have calculated the forward probabilities in $O(TC^2)$ time we Backwards-Sample using the following procedure,

$$z_i^{(T)} \sim \text{Discrete}(\alpha_{T,:}),$$
$$z_i^{(t)} \sim \text{Discrete}\left(\alpha_{t,:}\odot a_{:,z_i^{(t+1)}}\right), \quad t=T-1,\dots,0.$$

3. Experiments

3.1. Dataset

We used the Movielens dataset [1] to evaluate our models. Movielens offers several curated datasets of different sizes, 100k, 1M, 10M, 20M. For computational reasons, we chose the 100k dataset which contains 943 users and 1682 movies and 100k ratings on a 1-5 scale. Each user in the dataset has rated at least 20 movies.

3.2. Baselines

To evaluate our extension models we implemented 4 simple models as baselines and use root mean squared error (RMSE) as our metric. We use several of the same baselines described in [2]. The first one, MeanModel, simply predicts the empirical mean of observed ratings, i.e.

$$\hat{r}_{kl} = \frac{1}{S}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij} r_{ij},$$

where $S = \sum_{i=1}^N \sum_{j=1}^M I_{ij}$.

The second model, BiasModel, includes biases for each user and rating in addition to a global bias, i.e.

$$\hat{r}_{ij} = \alpha + \beta_i + \beta_j.$$

Table 1: RMSE for the models and the baselines.

	k	C	λ	Method	RMSE
MeanModel					1.154
BiasModel					0.955
LatentModel	30		1e-4		**0.931**
PMF	30		1.5e-4		0.946
MixtureModel	30	943	1.5e-4	EM	0.941
MixtureModel	30	471	1.5e-4	EM	0.940
MixtureModel	30	100	1.5e-4	EM	0.951
MixtureBPMF	30	100		Gibbs	**0.921**
TemporalModel	30	100		Gibbs	0.932

The third baseline, LatentModel, adds k-dimensional user and item latent feature vectors and is of the form

$$\hat{r}_{ij} = \alpha + \beta_i + \beta_j + u_i^T v_j.$$

We use L2 regularization for the latent features u_i and v_j. This is similar to the original PMF model with additional bias parameters. There are two hyperparameters to choose for this model, the latent dimension k and regularization strength λ.

The fourth baseline is the PMF as described above. We implemented the most basic version of PMF without either of the improvements suggested in [4]. We set $\lambda_U = \lambda_V$

The parameters of all baselines (except MeanModel) were trained using stochastic gradient descent.

4. Results

4.1. Mixture Model

For the mixture model we used EM to find a MAP estimate of the parameters and the posterior over the hidden states. We used $k = 30$ latent features and $\lambda = 10^{-4}$ L2 regularization strength. We trained the model with varying number of mixture components C. For $C = 943$ which is equal to the number of users we initialized the hidden mixture states with the identity matrix. Initializing the hidden states randomly made our optimization get stuck. Our predicted rating, $\hat{r}_{ij}$, of user i for movie j is the expected rating under the posterior-predictive distribution. The posterior is

$$P(\hat{r}_{ij}|R,U,V,\pi) = \sum_{c=1}^{C} \tilde{p}_c^{(i)} \mathcal{N}\left(\hat{r}_{ij}|u_c^T v_j, 1\right)$$

where $\tilde{p}^{(i)} = P(z_i|U,V,\pi,R)$. Our prediction then becomes

$$E\left[\hat{r}_{ij}|R,U,V,\pi\right] = \sum_{c=1}^{C} \tilde{p}_c^{(i)} u_c^T v_j$$

For models with $C < 943$ we initialized its mixture latent vectors by randomly picking them from the $C = 943$ model.

4.2. Mixture BPMF Model

Each sample collected for the Mixture BPMF model is for a full pass of the Gibbs sampler over all parameters. Prediction is done by taking the mean of the predictions for each sample. The sample mean approximates the posterior-predictive distribution. Figure 4 shows the RMSE as a function of epoch.

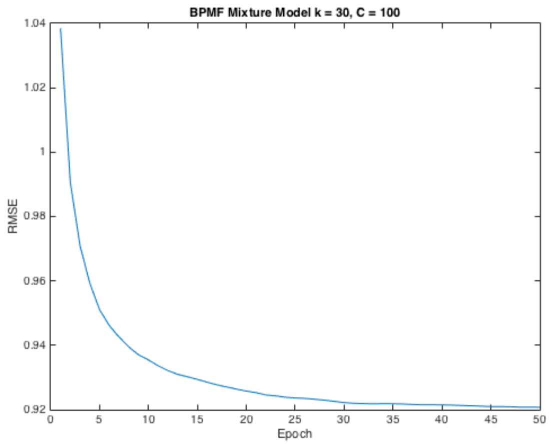

Figure 4: Mixture BMPF Model RMSE vs epoch

4.3. Temporal Model

The MovieLens dataset includes ratings from a period of 7 months, which we divided into 10 equally sized time steps. We again used $k = 30$ and $C = 100$.

4.4. Discussion

Table 1 shows the RMSE for all the models. We got the best results using the MixtureBPMF model. This is indicates that the temporal model may be too complex, while the MAP estimate found by the MixtureModel is too confident.

These results show that reducing the number of clusters to less than the number of users still gives reasonable results. This confirms our intuition that it is not really necessary to have a unique latent representation for each user. For example, reducing the number of clusters from 943 (i.e. one cluster per user) to 471 produces almost no change in RMSE. In other words, our model does not lose much predictive power by grouping users together. Even reducing the cluster number down to 100 still gave reasonable performance. We were primarily interested in how the models would do when forced to group many users together, which is why we chose such a small number of clusters.

Figure 5 shows the cluster assignments after the final time-step of the temporal model. Most of the users are assigned to one of 3 or 4 clusters while the remaining users are sparsely spread over the remaining clusters. This is an interesting result because it shows that user preferences are even more similar than we initially thought - i.e. most users can be well described using a very small number of latent profiles. This also indicates that the loss in predictive power from reducing the cluster number comes primarily from the inability to model outliers - i.e. users who would be assigned either to their own cluster or to a cluster with very few other users.

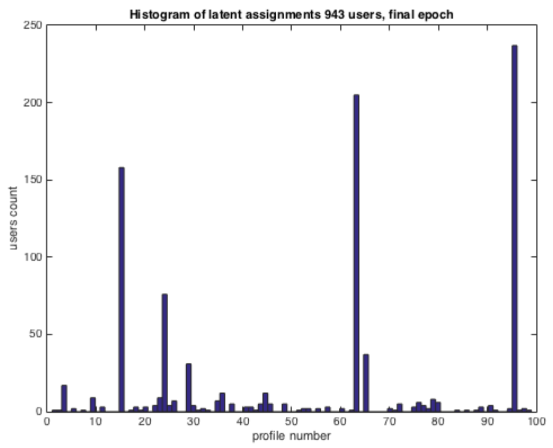

Figure 5: Histogram of user profile assignments

Figure 6 shows a temperature map of the transition probability matrix for the temporal model. The bright diagonal suggests that users tend to stay in the same cluster over time. This is especially true of the large clusters in Figure 5. Of course, users can still switch clusters if the observation gives a strong enough signal. But the tendency is to remain in the same cluster. This is likely a function of the data as much as a real effect. Most users rate relatively few movies, and by splitting them up into bins by time we make the data even more sparse. Of course, for users who rank a large number of movies this is less of a problem and the HMM may help to capture changes in these users' preferences over time. But for users with fewer ratings, the sparsity induced by time binning is more problematic.

This sparsity problem is likely one of the main reasons why the TemporalModel has a worse RMSE than the MixtureMPMF model. Clustering helps to reduce this problem (by assigning similar users to the same cluster, thus combining their rating information) but does not seem to completely solve it.

5. Conclusion

In this project, we explored several extensions to the original PMF model. In the first one, we replaced individual user vectors with user profiles (cluster). We hoped that

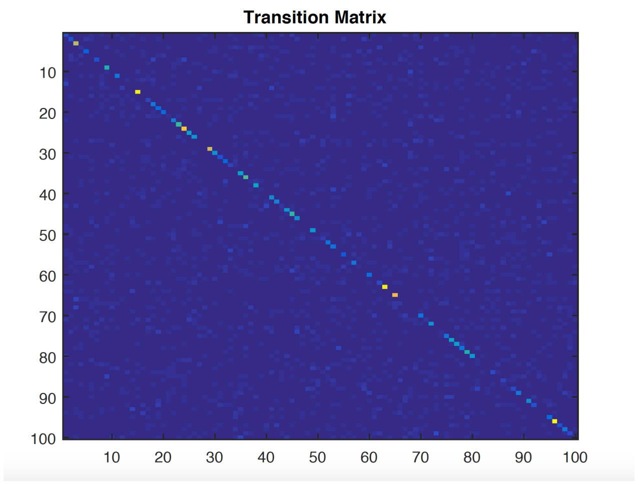

Figure 6: Temporal Model HMM State Transition Probabilities

doing so would improve scalability and generalization error (by helping with the cold start problem). We used both a MAP estimate of the model parameters and did a full posterior estimation using Gibbs sampling and added hyperpriors. Our Bayesian mixture model was able to improve upon the baselines.

Our second extension was adding Markov dynamics on the user profile assignments. We hoped that the combination of cluster assignments and temporal dynamics would allow the model to pick up on useful patterns in the data without overfitting. However this model did not improve on the best baseline, likely due to the sparsity of the data.

5.1. Future Work

There are several possible directions for future work. One interesting extension would be to do clustering on movies, in addition to users. Most movies are likely to fall in groups of roughly similar categories and extending the model in this way would be straightforward.

Another useful extension would be to try and deal with the sparsity of data induced by splitting the data into time bins. It is probably better to simply assign users with few ratings to a single cluster, since any cluster changes are probably just a result of noisy data. However, this is difficult to do in a principled way. One option would be to give each user (or cluster) their own transition matrix, with different priors depending on the number of movies rated. However, this would greatly increase the number of parameters to learn.

References

[1] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *ACM Transactions*

on Interactive Intelligent Systems (TiiS), 5(4):19, 2015.

[2] Yehuda Koren. The bellkor solution to the netflix grand prize. 2009.

[3] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[4] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. Citeseer, 2007.

[5] Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 880–887, New York, NY, USA, 2008. ACM.