Report on Visualization Project

Daiqi Linghu, Dunzhu Li, Yiran Ma

1 Introduction

In this project, the data we tackle with are the user ratings of the movies, and we aim to create two-dimensional illustrations of the latent factors. The formulation of the problem is:

$$Y_{M\times N} = U^T_{K\times M} V_{K\times N}$$

where M is the number of users, N is the number of movies, Y is the sparse rating matrix, U and V are the latent factors.

We first solve for U and V using Bias-Stochastic Gradient Descent (BSGD) method, and then, visualize U and V through PCA.

2 Algorithm

2.1 Optimization

To solve for the latent factors U and V, we solve the following optimization problem:

$$\underset{U,V,a,b}{\operatorname{argmin}} \frac{\lambda}{2}\left(\|U\|^2 + \|V\|^2 + \|a\|^2 + \|b\|^2\right) + \sum_{(i,j)\in S} (Q_{i,j})^2$$

where $Q_{i,j} = (u_i^T v_j + a_i + b_j + \mu - Y_{i,j})$. Then,

$$\frac{\partial J}{\partial U_l} = \lambda U_l + \sum_{(i,j)\in S} 2Q_{i,j} 1_{i=l} V_j$$

We call the first term in the gradient as "norm term", and the second term "error term".

U_l is updated iteratively through gradient method:

$$U_l^{n+1} = U_l^n - \eta \frac{\partial J}{\partial U_l}$$

In BSGD, for each iteration, we randomly loop through all the data and update U_l with "error term" only in the loop, then after the loop, correct U_l with "norm term". We can do V_k, a_l and b_k similarly.

2.2 Visualization

We project U and V to 2 dimensions and visualize them. First, we compute SVD of V:

$$V = A\Sigma B^T$$

The first two columns of A correspond to best 2-dimensional projection of movies V.

Then, we project every movie ($V_{1:N}$) and user ($U_{1:M}$) using $A_{1:2}$:

$$\tilde{V} = A_{1:2}^T V \in R^{2\times N}$$
$$\tilde{U} = A_{1:2}^T U \in R^{2\times N}$$

3 Implementation

3.1 Find U & V (BSGD.m)

To solve the optimization problem with Bias-SGD, we simply divide the data by 5 fold, use 4 folds as training data and 1 fold as test data to find the optimal damping parameter λ and number of iteration. The step size η is chosen as 1e-2 and is decreased by 10% for each iteration. From the results shown below (Figure 1), we choose $\lambda = 25$. For $\lambda = 25$, the out-of-sample error is still decreasing at the 100th iteration (Figure 1), however, the improvement is minimal. Therefore, we choose the number of iteration as 100.

3.2 Projection (Visualization.py)

To visualize the movie factor V, first we choose 101 movies which we are familiar with. We project them, and color them with their tags (category). Since there could be several tags for one movie in the original data, we re-tag them uniquely according to our own understanding. The result is shown in (Figure 2). We observe several patterns. As an example, we show the "Romance", "Sci-Fi" and "Crime" clusters in Figure 3. We also show several representative movie series ("Star Trek", "Free Willy", "Godfather", "Batman", "Terminator", and "Hepburn") in Figure 4. We also visualized different genres in the entire movie database, and several of them are showed in Figure 5.

The user factor is projected in Figure 6. Because we do not have any information about the users, we cannot discuss it further.

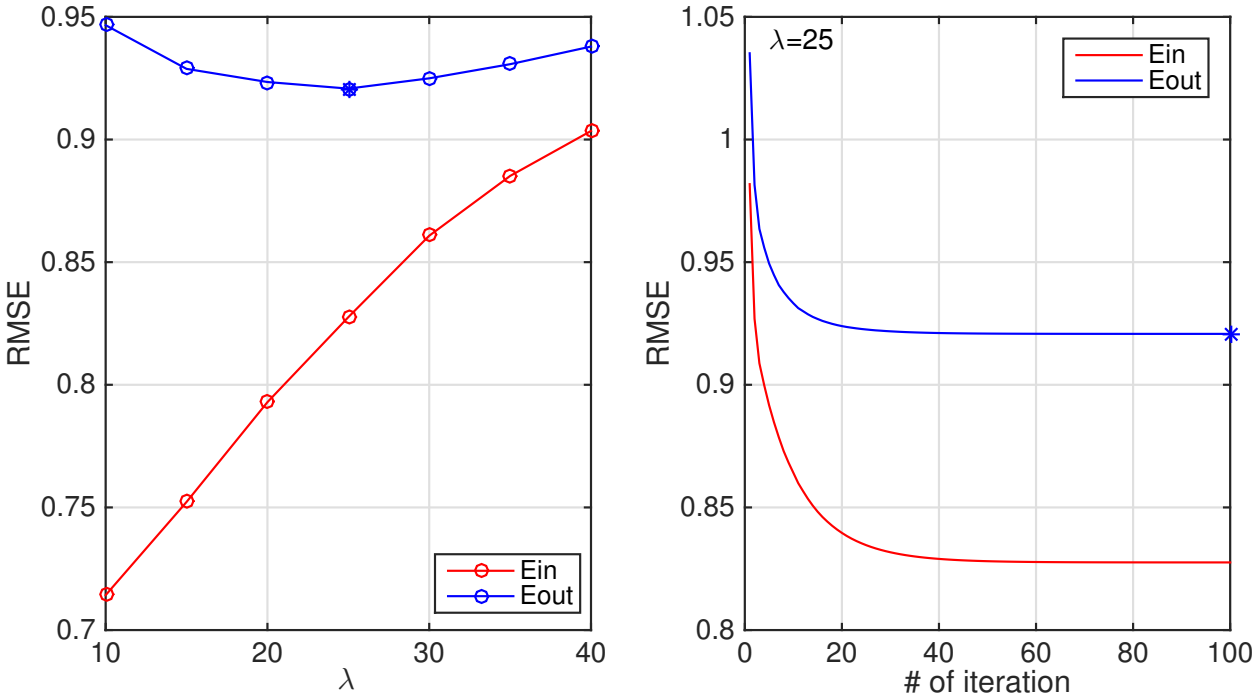


Figure 1: RMS error (RMSE) vs. Parameter. Left: For different λ, we plot the minimum RMSE in 100 iterations. We see that $\lambda = 25$ is the optimum choice. Right: For $\lambda = 25$, we plot the RMSEs in 100 iterations. We see that RMSE decays slowly after about 15 iterations. We simply choose the number of iteration as 100 for final run.

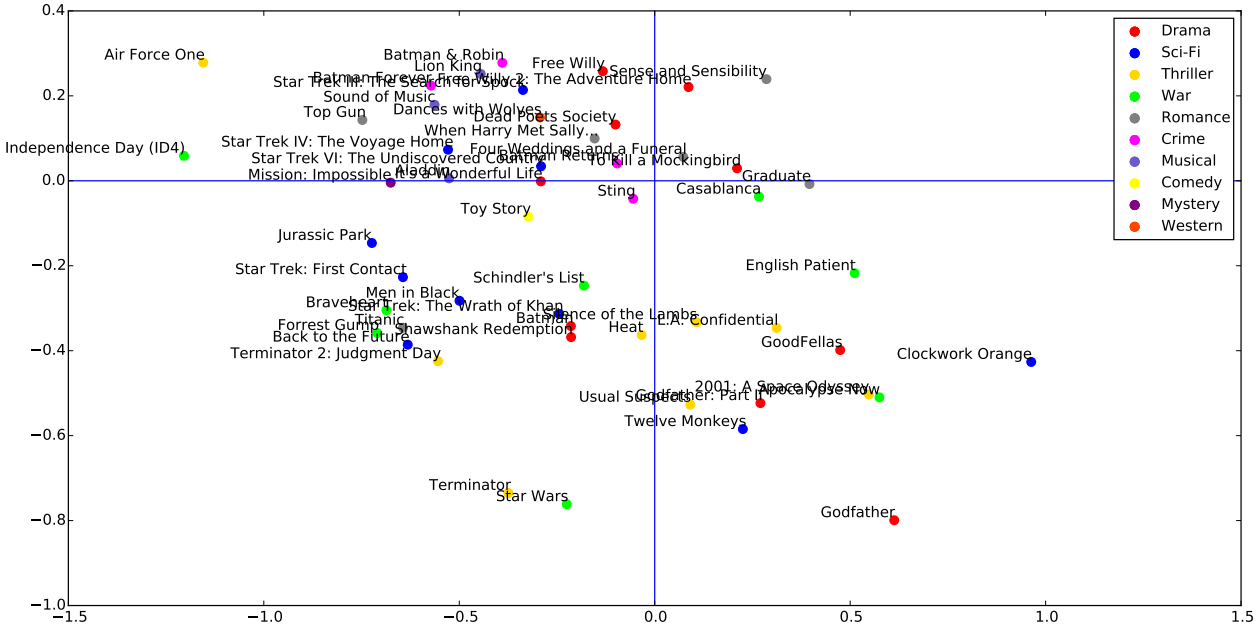


Figure 2: Projection of 101 selected movies.

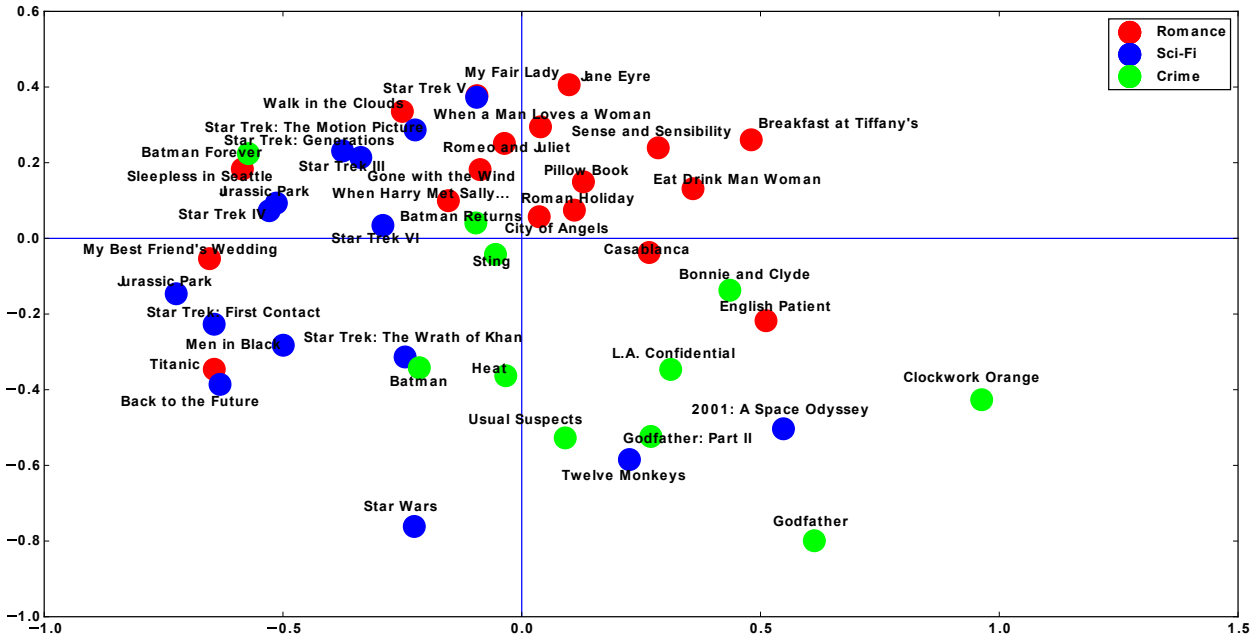


Figure 3: Projection of three representative movie catalogs.

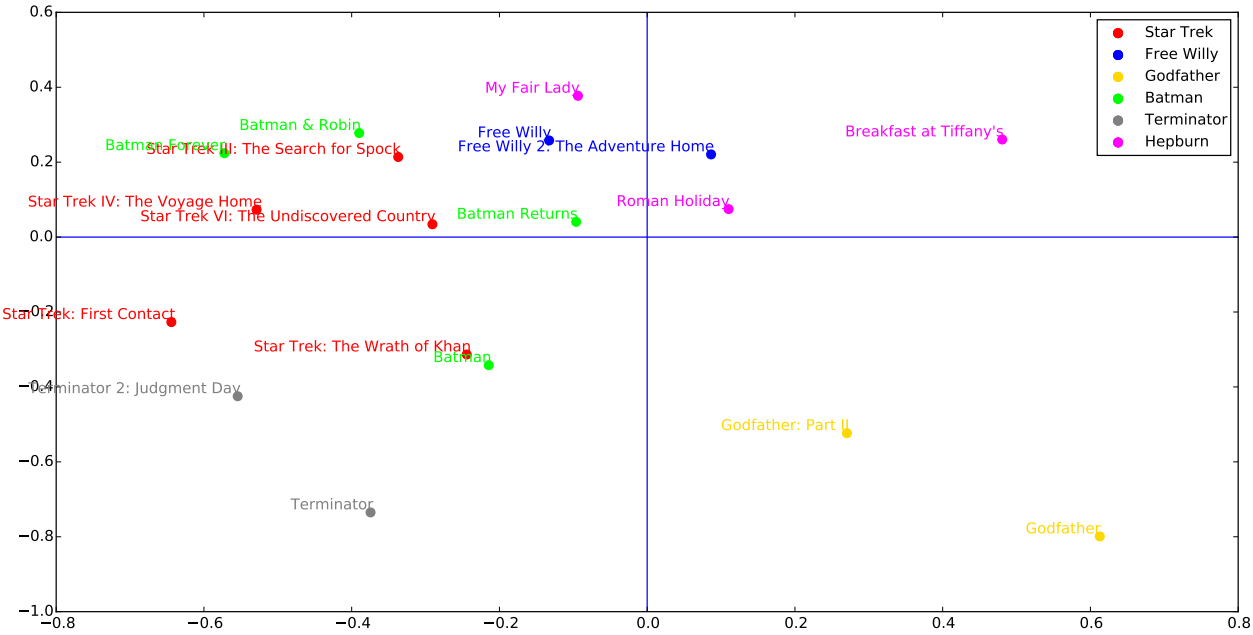
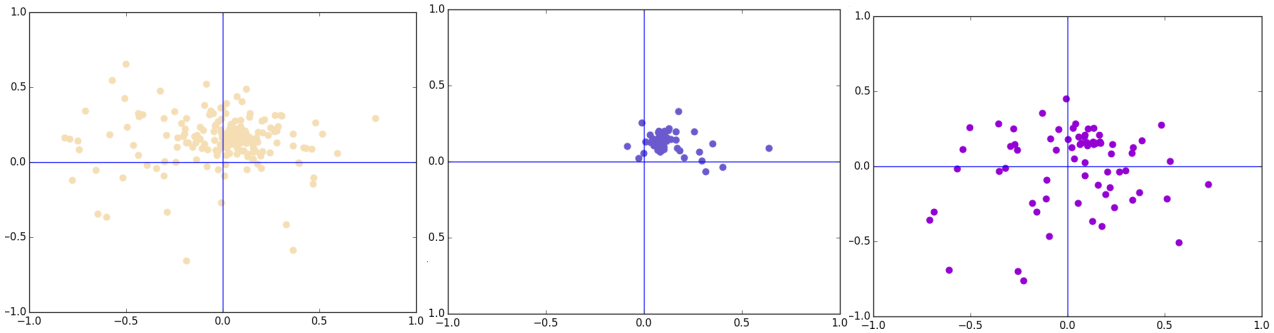


Figure 4: Projection of several movie series.



(a) Romance (b) Documentary (c) War

Figure 5

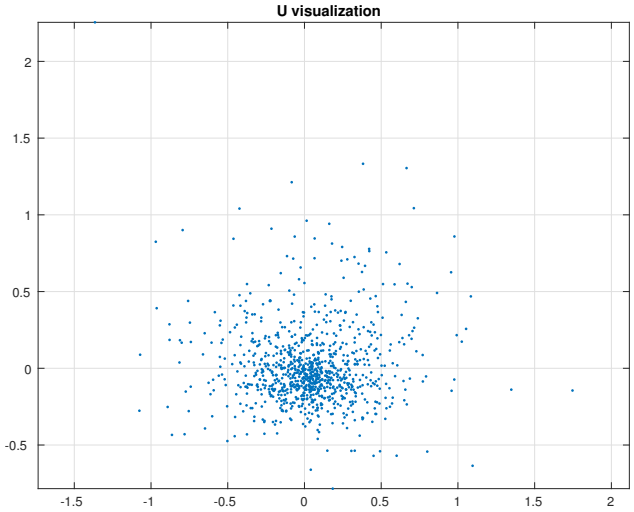


Figure 6: Projection of U

4 Discussions and Conclusions

From Figure 2 it's hard to find obvious patterns that can be easily interpreted by human, but we did manage to extract some useful information from it. Figure 3 clearly shows that the "Romance", "Sci-Fi" and "Crime" movies formed three clusters which can (more or less) be separated from each other. What's more interesting is that the three genres were picked by three team members respectively, which means our different movie preference can indeed be visualized. The pattern is much more clear compared to Figure 5, which uses all the data, including popular movies and movies that few people ever rated, and hence has much more noise and gives less reliable results.

From Figure 4 shows that movies in the same series are relatively close to each other on the 2-D plot, which make sense since they're usually shot by the same group of people, and have similar stories to tell. Moreover, the movies starring the same actor can be close to each other on the 2-D plot as well, and in Figure 4 we took Audrey Hepburn as an example.

We can't say too much about the user (see Figure 6) since the only thing we know about them are the ratings they gave. One way to work on the user projection is to cluster them, and try to extract information out of it (i.e. Children or Adult? Male or Female?). However since we don't have the labels, all the conjectures can not be proved or disapproved. Although we can find the users which lie closely to "Romance" movies on the plot like "Romance" movies, but the users which lie further in the direction like "Romance" movies even more so the visualization won't help much. We didn't seek to do more based on Figure 6.

To summarize, yes, we can visualize SVD algorithm and draw some useful basic conclusions. However, since each movie is only represented by two values in the plot, the signal/noise ratio is relatively low, and we shouldn't try to over-interpret (over-fit) the visualization.