Incorporating Matrix Factorization Algorithms in an Anime Recommendation System: A Comparative Study

Fay Feghali
Khoury College of Computer Science
Northeastern University, U.S.A
Feghali.fa@northeastern.edu

Abstract

User-based collaborative filtering (UCF) techniques are commonly used for recommendation systems (RS). There are many different approaches to UCF, and one of the most common methods is matrix factorization (MF). In this paper, we will be comparing three different matrix factorization techniques: singular value decomposition (SVD), alternating least squares (ALS) and stochastic gradient descent (SGD). Each algorithm will be applied to the same dataset and will be compared using root mean square error (RMSE). Our preliminary results show that MF technique that used SGD performs better than the other two MF algorithms.

1 Introduction

MyAnimeList (MAL) is the largest online forum for English-speaking anime enthusiasts to rank and discuss their favorite shows. With 150 million page views and over twelve million unique visitors each month, MAL consistently gathers a plethora of data. Since users can request to add shows and movies that are not currently on the website, there are likely very few shows that do not exist as a data point on the site. Even with this enormous user-sourced data set, the recommendations section of MAL leaves much to be desired. Like most everything on the site, recommendations are user generated. MAL lacks a recommendation engine that utilizes its ever-growing user base. This project aims to address this issue by evaluating three different recommendation algorithms on our MAL dataset and then applying those techniques to develop a recommendation system (RS) that MAL users can use to find new anime to watch.

There are two general approaches to RSs. Content-based filtering is an approach that requires creating a profile for each user or item that characterizes that user or item. For example, users on a shopping site may be categorized based on their demographics and what they choose to purchase. An item on a shopping website could be categorized by how popular it is and what department it is in. In content-based filtering, users and items are matched using such profiles. The second strategy for RSs is the collaborative filtering (CF) approach. CF relies on the relationships between users and items that are determined by calculating similarities like latent factors. These factors are less well-defined dimensions of a dataset that are found using latent factor models.

Matrix factorization (MF) is a common approach for latent factor modeling. In a famous case, the Netflix Prize Problem, there were breakthrough analyses on matrix factorization algorithms used on movie data. Three MF models examined in this paper were singular value decomposition (SVD), stochastic gradient descent (SDG), and alternating least squares (ALS) [i]. In RSs, these techniques are used to decompose user-item-rating matrices to predict user's ratings for items. These predictions are then used to recommend items to users.

Motivated by these previous analyses on RSs and MF, in this report we propose to compare these three MF algorithms using a MAL dataset and present our evaluation of those results. The root mean square error (RMSE) is used to evaluate the performance of the models and cosine similarity is used in combination with the MF methods to output the top ten recommendations for a user-given anime title.

Overview of Paper We first summarize background information needed to understand each MF approach. In section 3, we delve into RS related work. In the next section, we describe the formal details of the MF algorithms used. In section 5, we present and discuss our evaluation results from the experiments. The last section concludes our report by summarizing our results and looking towards the future.

2 Background

Matrix factorization models map users and items such that the user-item interactions, like ratings, are modeled as an inner product of the dataset. Each item is associated with vector and each user is associated with a vector. The dot product

of those two vectors approximates the user ratings for the items in the dataset. Obtaining the right mapping that gives these predicative ratings is what differentiates MF algorithms. In this report, we compare three MF techniques: SVD, SDG and ALS. These methods can be summarized as follows:

Singular Value Decomposition

SVD is characterized as a basic MF model known for identifying latent semantic factors for data information and retrieval. It decomposes the user-item ratings matrix into three submatrices. A combination of the dot product of those matrices is what gives predicative ratings.

Stochastic Gradient Descent

SGD is a machine learning algorithm that loops through all user's ratings of each item in a training set and predicts new ratings for unrated items based on the prediction error.

Alternating Least Squares

ALS is a learning algorithm that alternates between user and item vectors to make predictions. ALS has been utilized mainly for implicit datasets. We are using ALS particularly because it used by Crunchyroll, an anime streaming service, to recommend shows to its customers. Even though our data is not implicit, we want to look at how ALS performs on explicit data.

3 Related Work

Recommendation systems are typically categorized as either content-based recommendations or collaborative filtering. CB recommendations use user and item profiles to extract similarity data and suggest items to users. This approach tends to be extraction heavy and relies on gathering external information that may be difficult to obtain[ii]. CF has many successful implementations in commercial RSs. Amazon, Netflix, eBay and many other companies utilize CF algorithms to recommend items to users. These companies rely on RSs to keep their customers interacting and consuming their products. CF offers a more personalized recommendation approach than CB and is now widely studied in the RS research field.

CF-RSs on explicit datasets, like data that include user ratings, are usually sparse since majority of users tend to only rate a small number of items in a dataset. This sparsity can cause issues with CF techniques like nearest neighbors and clustering. In Koren *et al.*[iii], it is shown that MF algorithms yield better results with sparse data. A benefit to using MF is the ability incorporate implicit feedback from the user unlike CF techniques. MF also offers the benefit of inferring latent information about the data. Unfortunately, we do not have access to implicit data from MAL. It is unclear whether data is gathered by MAL on what users click on but nevertheless, MF models are successful at determining underlying features of sparse datasets and should yield better performance results in comparison to nearest neighbor algorithms like k-nearest-neighbors or clustering algorithms like k-means.

4 Approach

In this section, we will present the equations and parameters used to calculate the three MF methods as well as our approach to evaluating performance and presenting recommendations.

Singular Value Decomposition

The decomposition of the user-item-ratings matrix ($Data_{mxn}$) is calculated using the numpy's library SVD implementation. The dot product of the decomposed matrices is then calculated to find the predicative ratings for all the user-item pairs. SVD decomposition can be characterized by the following:

$$Data_{mxn} = U_{mxm} \Sigma_{mxn} V^T_{nxn}$$

where the U is a mxm matrix, $\sum$ is a mxn matrix that all values are 0 except for those lying on the main diagonal and all values in the diagonal line are called singular values. V is a nxn matrix, and VT represents the transpose matrix of V. U represents the users and V represents items. new anime to watch.

Stochastic Gradient Descent

The SGD algorithm factorizes a matrix similarly to SVD but instead iterates through all the ratings and predicts the ratings r by computing an associated predicted error. It then modifies the parameters by a magnitude proportional to the regularization constant in the opposite direction of the gradient which yields:

$$q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$$
$$p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$$

Alternating Least Squares

The ALS algorithm factorizes a given matrix R into two factors U and V such that $R \approx U^T$. The unknown row dimension is given as a parameter to the algorithm and represents latent factors. Matrix U represents users, matrix V represents items and matrix R is the ratings matrix. Finding R

using U and V requires an interactive approach represented by the following problem:

$$arg\ min_{U,V} \sum_{\{i,j \mid r_{i,j} \neq 0\}} (r_{i,j} - u_i^T v_j)^2 + \lambda \left(\sum_i n_{u_i} \|u_i\|^2 + \sum_j n_{v_j} \|v_j\|^2 \right)$$

Evaluation

Each algorithm will be evaluating by calculating the RMSE of the training set and the test set. The parameters in which they are compared is the latent factors. For ALS and SGD, the number of iterations is also compared for the training and test sets. The training and test sets were determined by taking out ten random rating from the original set and making that the training set while the test set just has the ten left out ratings.

Recommendations

After optimizing for the best models for each algorithm, the cosine similarity of the item latent vectors is used to calculate the similarity between the different anime. The input for this is an anime title that is available in our dataset and the output is the top ten anime that are most similar to the input anime title.

5 Experiments and Results

Data Filtering Although MAL has an extensive dataset, there are some drawbacks to study MF methods. There are many anime titles that are only rated once or users that have only rated one anime. Because of this, the data needs to be filtered. After taking out titles and users with a total rating count less than 1000 and 750 respectively, we were left with a much cleaner dataset. The sparsity of our dataset was 32.79%.

Datasets	Users	Anime	Ratings
Before Cleaning	106,363	6598	19,171,950
After Cleaning	2124	3043	2,119,516

Table 1: The Dataset of Anime Ratings

Latent Factors Analysis In this section we present data on the latent factor analysis for all three algorithms. Each algorithm is run 8 times using a different number of latent factors for each run. The number of latent factors is 10, 20, 30, 40, 50, 60, 70, and 80. The best model for each algorithm is determined by calculating the RMSE for each run. The best model for SVD is 50 factors with a RMSE of 1.12 for the training set and 1.22 for the test set. The best model for SGD is 20 factors with a RMSE of 1.13 for both the sets. The best model for ALS is 60 factors with a RMSE of 3.12 for the training set and 3.56 for the test set.

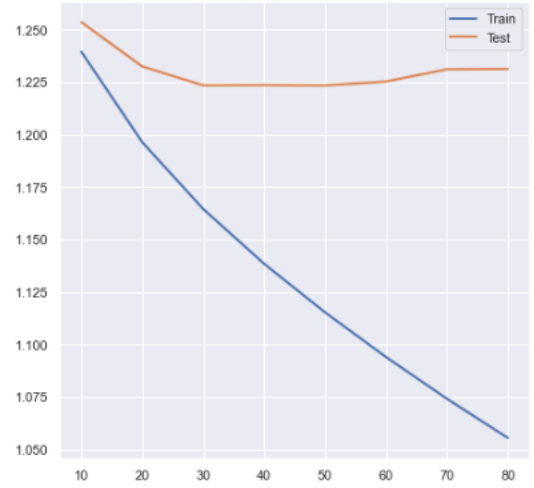

Figure 1: SVD Latent Factors versus RMSE

Figure 2: SGD Latent Factors versus RMSE

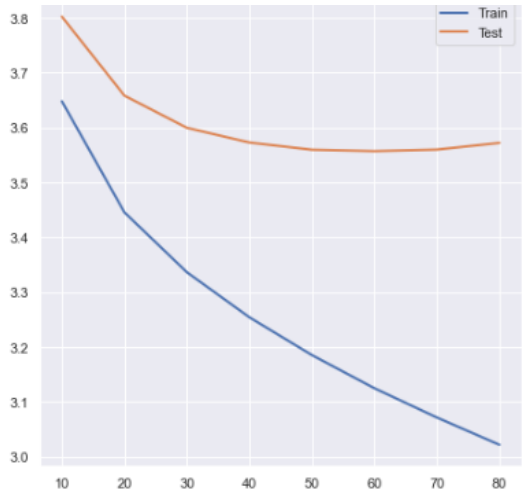

Figure 3: ALS Latent Factors versus RMSE

The SVD model shows overfitting and irregular pattern comparing the train and test set. The best model out of the three based on the RMSE is SVD by only 0.01 comparing to the RMSE for SGD. However, since SGD has a closer fit than SGD, we consider SGD the better model for this dataset.

Learning Curve Analysis In this section we present data on the learning curve analysis comparing the two machine learning algorithms in this report, SGD and ALS. Each algorithm was tested for 6 different iterations [1, 2, 5, 10, 25, 50, 100]. The best model for each algorithm is determined by calculating the RMSE for each iteration in the iteration array. The best model for SGD is 100 iterations with a RMSE of 1.13 for both the sets. The best model for ALS is 50 factors with a RMSE of 3.12 for the training set and 3.56 for the test set.

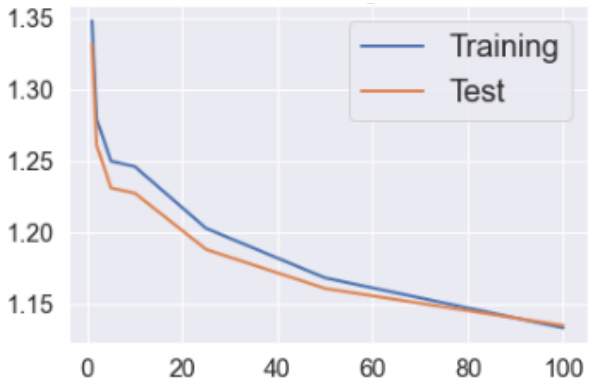

Figure 4: ALS Learning Curve Iterations versus RMSE

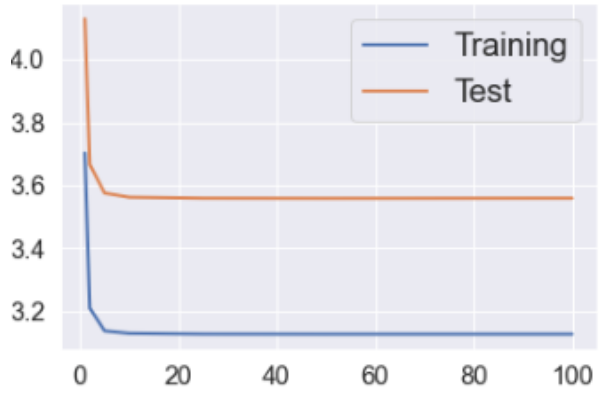

Figure 5: ALS Learning Curve: Iterations versus RMSE

The SDG model shows the least overfitting while the ALS model has a lot of overfitting.

Recommendations In this section, we present the top ten recommendations for all three algorithms. Using cosine similarity we are able to calculate the top ten anime that are most similar to 'One Punch Man' for each model.

Rank	Anime Title
1	One Piece: Umi no Heso no Daibouken-hen
2	One Piece Movie 4: Dead End no Bouken
3	One Piece Movie 5: Norowareta Seiken
4	Oni Chichi: Re-birth
5	One Punch Man Specials
6	One Piece: Straw Hat Theater
7	Ookiku Furikabutte
8	One Piece: Heart of Gold
9	One Piece: Glorious Island
10	One Piece: Yume no Soccer Ou!

Table 2: Top Ten Recommendations for 'One Punch Man' using SVD

Rank	Anime Title
1	Nanatsu no Taizai
2	One Punch Man: Road to Hero
3	Mob Psycho 100
4	No Game No Life
5	Boku no Hero Academia
6	One Punch Man Specials
7	Boku no Hero Academia 2nd Season
8	Katanagatari
9	Bakemono no Ko
10	One Piece Film: Gold

Table 3: Top Ten Recommendations for 'One Punch Man' using SGD

Rank	Anime Title
1	Boku no Hero Academia
2	Shingeki no Kyojin
3	Re:Zero kara Hajimeru Isekai Seikatsu
4	Overlord
5	No Game No Life
6	Kono Subarashii Sekai ni Shukufuku wo!
7	Nanatsu no Taizai
8	Boku dake ga Inai Machi
9	Shokugeki no Souma
10	Dungeon ni Deai wo Motomeru no wa Machigatteiru Darou ka

Table 4: Top Ten Recommendations for 'One Punch Man' using ALS

Conclusion

In conclusion, SGD performed the best out of the three algorithms. Given these results, MAL should use the recommendations given by the SGD model with 20 latent factors and 100 iterations. The only drawback to this is that the SGD performance time was a lot more than both SVD and ALS. Running the SVD performance metrics took about five times longer than ALS and SVD. This may be due to the limitations of the machines that we are using as well as the number of biases considered for the SGD algorithm versus the other two MF algorithms in this report. The incorporation of more bias factors and regularization did however improve the model as it was the best fitting model when looking at the training and test sets.

In the future we hope to improve this study by incorporating more regularization techniques to help reduce overfitting. Also, we will investigate combining different CF algorithms to make hybrid recommendation systems. The original dataset can also be improved by taking a more current dataset instead of one from four years ago. Improving on these issues will surely give an even better recommendation engine for MyAnimeList.

References

Advertising. In: MyAnimeList.net. https://myanimelist.net/advertising. Accessed 1 May 2022

Bhattacharyya M (2020) Beginner's Guide to creating an SVD recommender system. In: Medium. https://towardsdatascience.com/beginners-guide-to-creating-an-svd-recommender-system-1fd7326d1f65. Accessed 3 May 2022

Myanimelist dataset. In: Kaggle. https://www.kaggle.com/datasets/azathoth42/myanimelist. Accessed 2 May 2022

Numpy.linalg.svd. In: numpy.linalg.svd - NumPy v1.22 Manual. https://numpy.org/doc/stable/reference/generated/numpy.linalg.svd.html. Accessed 3 May 2022

Rosenthal E (2016) Explicit matrix factorization: ALS, SGD, and all that jazz. In: Medium. https://blog.insightdatascience.com/explicit-matrix-factorization-als-sgd-and-all-that-jazz-b00e4d9b21ea. Accessed 3 May 2022

Vito Xituo Chen Dept. of Computer Science University of Southern California, Chen VX, Dept. of Computer Science University of Southern California, et al (2019). Incorporating singular value decomposition in user-based collaborative filtering technique for a movie recommendation system: In: ACM Other conferences. https://dl.acm.org/doi/10.1145/3357777.3357782. Accessed 25 April 2022

Xiaoyuan Su Department of Computer Science and Engineering, Su X, Department of Computer Science and Engineering, et al (2009) A survey of collaborative filtering techniques. In: Advances in Artificial Intelligence. https://dl.acm.org/doi/10.1155/2009/421425. Accessed 26 April 2022

Y. Koren, R. Bell and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," in *Computer*, vol. 42, no. 8, pp. 30-37, Aug. 2009, doi: 10.1109/MC.2009.263.

Y. Li, J. Hu, C. Zhai, and Y. Chen. 2010. Improving One- Class Collaborative Filtering by Incorporating Rich User Information. 19th ACM international conference on Information and knowledge management, 959-968. ACM.