Deep Maximum Margin Matrix Factorization for Collaborative Filtering based Recommender Systems

Gaurav Niranjan[a], Venkateswara Rao Kagita[b,*], Vikas Kumar[c,*], Adamya Shyam[c]

[a] *The LNM Institute of Information Technology*
[b] *National Institute of Technology, Warangal, India*
[c] *University of Delhi, Delhi, India*

Abstract

Collaborative filtering research has found success with the learning technique known as Maximum Margin Matrix Factorization (MMMF). In order to simultaneously approximate the observed entries under a partially observed ordinal rating matrix, low-norm latent component matrices U (of users) and V (of items) must be determined. Unobserved entries are measured and predicted by some loss. Since there are just two levels (± 1) in the rating matrix, rows of V can be thought of as points in k-dimensional space and rows of U as decision hyperplanes dividing +1 entries from -1 entries. When the loss function is hinge/smooth hinge loss, the hyperplanes serve as maximum-margin separators. When using Deep MMMF, we employ a neural network architecture to perform the matrix factorization task and the hinge loss function is specifically extended to accommodate several levels in order to handle rating matrices with multiple discrete values. Along with learning user and item latent vectors, we additionally learn another vector θ, unique for each user, that seperates the user-item interaction hyperplane into several parts depending upon the number of possible ratings. Using two benchmark datasets, we compare our method with ten well-known algorithms and demonstrate that it outperforms them on the NMAE metric.

Keywords: Collaborative Filtering, Recommender Systems, Deep Learning, Matrix Factorization

1. Introduction

Recommender systems improve the overall user experience, boost user interaction, and contribute to sales growth and user satisfaction by delivering customized and pertinent recommendations. Collaborative filtering is one of the successful approaches of recommender systems that capture the patterns of other users' profiles and use those patterns to recommend

*Corresponding author
Email addresses: 19ucs054@lnmiit.ac.in (Gaurav Niranjan), venkat.kagita@nitw.ac.in (Venkateswara Rao Kagita), vikas007bca@gmail.com (Vikas Kumar), ashyam@cs.du.ac.in (Adamya Shyam)

items suitable to the target user profile. Matrix factorization (MF) is a collaborative filtering technique that decomposes user-item rating matrix Y into two low-rank matrices, U and V, called user and item latent factor matrices, respectively. Matrix factorization aims to find low-dimensional latent factors that can effectively capture the underlying patterns and relationships in the user-item matrix. Decomposing the matrix into lower-rank matrices makes it possible to represent users and items in a latent space, thereby allowing for a more accurate modeling of interactions between users and items. MF techniques learn user and item latent factor matrices that accurately approximate all the observed entries in a partially observed rating matrix. This process simultaneously approximates the unobserved entries and completes the matrix. Traditional methods on matrix factorization includes Nonnegative Matrix Factorization [Lee & Seung (1999)], SVD and its several variations like SVD++ [Koren (2008)], Asymmetric SVD [Paterek (2007)] and Group-specific SVD [Bi et al. (2017)].

Maximum Margin Matrix Factorization (MMMF) Srebro et al. (2004) gives a unique geometrical interpretation of the matrix factorization. It considers the items' latent factors as points in the k dimensional space, and user latent factors are the hyperplanes that separate items with different ratings. However, the problem with this approach is it tries to project the items' latent factors onto a k dimensional subspace so that the various ratings are separable from the parallel lines. The constraint of parallel lines makes the learning problem very restrictive or cumbersome and can only model linear interactions between the users and items. This limitation of MMMF motivates the development of an approach that can model nonlinear interactions between users and items. Precisely, we investigate developing a method that learns nonlinear decision boundaries.

We propose a deep maximum margin matrix factorization architecture (deep MMMF) that models the nonlinear interactions between the users and items. The proposed model transforms the problem through a series of deep neural layers that induces nonlinearity into the model. We demonstrate that the MMMF approach cannot learn a separating hyperplane for complex cases and how deep MMMF learns separating hyperplanes to distinguish the different ratings of users. We perform extensive experiments on real-world datasets and show that the proposed approach outperforms MMMF and other state-of-the-order methods.

The rest of the paper is structured as follows. Section 2 summarizes the state-of-the-art literature. We describe the proposed approach in Section 3. Section 4 presents the experimental analysis and results. We conclude with Section 5.

2. Literature review

Deep learning techniques have shown remarkable success in learning hierarchical representations from raw data, driving transformative advancements in the domain of information retrieval. Concurrently, the efficacy of matrix factorization methods in revealing latent relationships from data has been exhibited through applications in recommendation systems and collaborative filtering. Recently, several works have been done on deep learning-based matrix factorization techniques to leverage the capacity of deep learning models and matrix factorization together to reveal latent structures within complex datasets. In this section, we discuss the existing approaches that explore the fusion of deep learning and matrix factorization paradigms, highlighting the evolution of this hybrid methodology and the novel insights it has brought to various application domains.

By preemptively comprehending user-item dynamics, Fu et al. (2018) acquire embeddings for users and items separately to capture semantic correlations and further train a feed-forward neural network on the pre-trained embeddings to replicate user-item interactions, fostering accurate predictions. In a three staged sequential approach, Bobadilla et al. (2020) utilizes matrix factorization at first stage, and a multilayer neural network at the remaining two stages to exploit the potential of real prediction errors obtained from first stage, and enhance the recommendation quality. In Lara-Cabrera et al. (2020), authors harnessed the power of deep learning techniques by treating matrix factorization as multiple layers, such that the output of one layer serves as the input of the subsequent layer. Taking the interaction matrix as an input to a neural network, Xue et al. (2017) learn the low-dimensional representation of users and items. The model encompasses both explicit ratings as well as non-preference feedback, thus maximizing the information extracted.

In He et al. (2017), a novel generic framework for neural collaborative filtering (NCF) is devised, where the inner product is replaced with a neural architecture to learn an arbitrary function from data. The proposed framework can be used with any deep learning model to provide recommendations. In another work, Yi et al. (2019) propose a generic deep matrix factorization (DMF) framework that enhances the recommendation accuracy by incorporating side information using two channel structure. DMF learns the low-dimensional embeddings of users and items directly using a feature transforming function, instead of from the observed data. They propose implicit feedback embedding (IFE) to transform users and items within an implicit feedback information graph to real-valued low-dimensional vectors, thus reducing the numbers of parameters and increasing training efficiency. With the help of convolutional neural networks and gated recurrent neural networks, Wu et al. (2018) develop a dual-regularized multilayered deep matrix factorization model to learn the user and item representations using text descriptions.

The fusion of neural network architectures with matrix factorization have significantly enriched the recommendation accuracy by learning intricate representations from data Da'u & Salim (2020); Zhang et al. (2019); Liu et al. (2022). In our work, we focus on to learn the nonlinear user-item interactions and develop a deep MMMF model that enhances the recommendation accuracy.

3. Proposed Approach

Matrix factorization (MF) is a prevalent collaborative filtering (CF) method, that leverages on the inherent patterns derived from the user-item interactions to generate precise and efficient automated recommendations. Given a user-item rating matrix $Y \in \mathbb{R}^{M\times N}$, wherein the entry $Y_{ij} \in \{0, 1, \ldots R\}$ indicates the rating given by a user i for an item j, R is the maximum possible rating, and 0 indicates an unobserved rating, the task is to predict the unknown ratings of this partially observed rating matrix. The matrix factorization approaches find users' latent factors U and items' latent factors S, so the interaction between the users and items can be determined through $UV^T \approx Y$ over all the observed entries. These user and item latent factors can represent meaningful patterns, features, or concepts that are not directly observable in the original matrix. They can effectively predict the unobserved ratings by leveraging the observed entries to estimate the missing values based on the factorized representations.

One of the prominent MF algorithm, the maximum margin matrix factorization (MMMF) provides a rich geometrical interpretation to the user and item latent features. It assumes that item latent features V are the points in the k-dimensional space and user latent features U are the hyperplanes that separate the instances belonging to the different ratings of a user. Each user hyperplanes separate the data differently based on their preferences. For each user, it learns $R-1$ separating hyperplanes that distinguish the items of R different ratings, where R is the maximum possible rating. Assuming that these lines are parallel, we need to determine $R-1$ different intercept values (θ's) alongside user latent factors as coefficient values of a hyperplane. Thus, the MMMF approach aims to learn the optimizing hyperplanes that maximize the margin. The objective of Maximum Margin Matrix Factorization is formulated as follows.

$$\min_{U,V,\theta} J(U,V,\theta) = \sum_{(i,j)\in\Omega} \sum_{r=1}^{L-1} h(T_{ij}^r(\theta_{i,r} - U_i V_j^T)) + \frac{\lambda}{2}(\|U\|_F^2 + \|V\|_F^2 + \|\theta\|_F^2), \tag{1}$$

$$J(U,V,\theta) = \sum_{(i,j)\in\Omega} \sum_{r=1}^{R-1} h(T_{ij}^r(\theta_{ir} - U_i V_j^T)) + \frac{\lambda}{2}(\|W_\theta\|_F^2 + \|W_U\|_F^2 + \|W_V\|_F^2),$$

where U_i and V_j are latent vector representations of user i and item j respectively. θ_{ir} is the threshold for rating r of user i. $\|.\|_F$ is the Frobenius norm, $\lambda > 0$ is regularization parameter, Ω is the set of observed entries, $h(.)$ is smoothed hinge loss as proposed in Rennie & Srebro (2005) and is defined as:

$$h(z) = \begin{cases} 0, & \text{if } z \geq 1, \\ \frac{1}{2}(1-z)^2, & \text{if } 0 < z < 1, \\ \frac{1}{2} - z, & \text{otherwise} \end{cases}$$

T is defined as follows:

$$T_{ij}^r = \begin{cases} +1, & \text{if } r \geq y_{ij}, \\ -1, & \text{if } r < y_{ij} \end{cases}$$

T can be interpreted as a matrix created from the matrix of all the observed entries with rating y_{ij} for user i given to item j such that matrix T follows the above rule for a fixed r. T makes the cost function a variation of R binary class problems for our multi-class problem at hand.

Geometrically, each V_j is a point and each (U_i, θ_{ir}) defines a decision hyperplane that divides the points that user i has rated a r. For example, when $R = 5$, 4 hyperplanes divide the entire space into 5 regions, each corresponding to one of the five ratings.

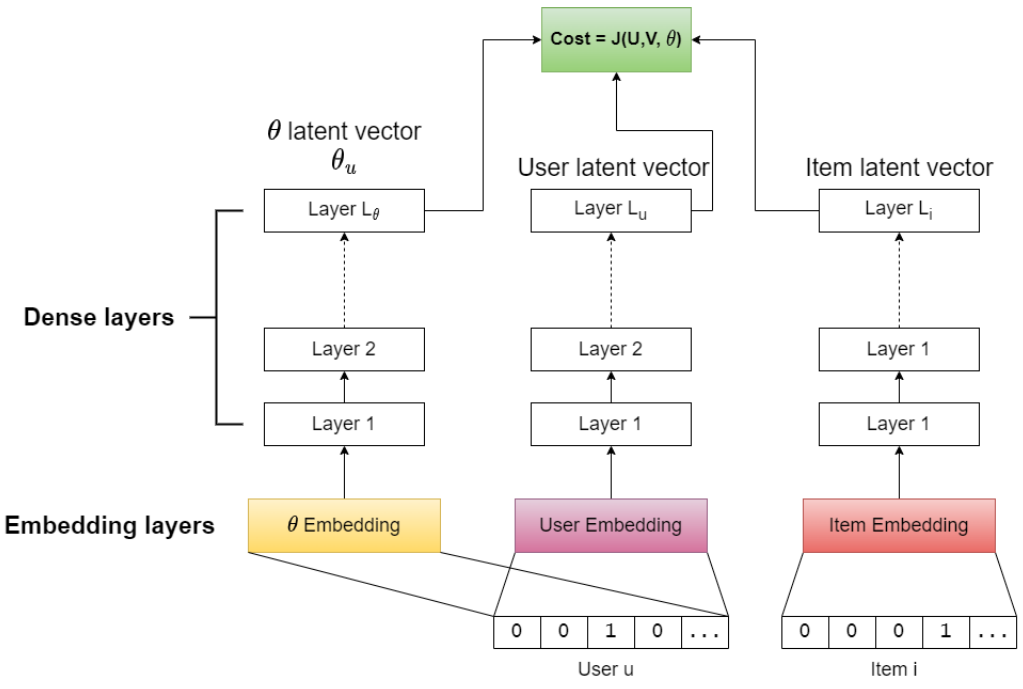


Figure 1: Illustration of the proposed neural network architecture

The MMMF approach implicitly compels two assumptions, i) the decision boundaries are hyperplanes and ii) these hyperplanes are parallel to one another, making the optimization problem more rigid and complex. This paper proposes a deep MMMF architecture that models the nonlinear interactions between the users and the items. Nonlinearity is induced into the model by passing the U, V, and θ through a series of deep neural network layers with nonlinear activation functions. Figure 1 demonstrates the proposed architecture, and the cost function of MMMF can now be rewritten as follows.

$$\min_{U,V,\theta} J(U,V,\theta) = \sum_{(i,j)\in\Omega} \sum_{r=1}^{L-1} h(T_{ij}^{r}(\varphi_1(\theta_{i,r}) - \varphi_2(U_i)\varphi_3(V_j^T))) + \frac{\lambda}{2}(\|U\|_F^2 + \|V\|_F^2 + \|\theta\|_F^2), \tag{2}$$

$$J(U,V,\theta) = \sum_{(i,j)\in\Omega} \sum_{r=1}^{R-1} h(T_{ij}^{r}(\varphi_1(\theta_{ir}) - \varphi_2(U_i)\varphi_3(V_j^T))) + \\ \frac{\lambda}{2}(\|W_\theta\|_F^2 + \|W_U\|_F^2 + \|W_V\|_F^2),$$

where $\varphi_1, \varphi_2, \varphi_3$ performs the nonlinear transformations on U, V, and θ, respectively.

4. Empirical Results

This section presents the empirical study of the proposed approach and a comparative analysis of the proposed method with the various state-of-the-art techniques on benchmark datasets.

4.1. Datasets

We use two benchmark datasets for our experiments: MovieLens 1M[1] and EachMovie[2]. The MovieLens dataset consists of 1,000,209 ratings given by 6,040 users for 3,952 movies, of which 3,706 movies are actually rated, and every user has at least 20 ratings. There are five possible rating values, $\{1, 2, .., 5\}$. The EachMovie dataset consists of 2,811,983 ratings given by 72,916 users for 1,628 movies, out of which 1,623 movies are rated are rated by atleast one user, and 36,656 users have given at least 20 ratings. There are six possible rating values, $\{0, 0.2, .., 1\}$ which we have mapped to $\{1, 2, .., 6\}$.

4.2. Experimental setup

We conduct our experiments with two standard setups in the recommender systems area.

1. *Weak generalization:* In the case of a weak generalization setup, the test set is produced by randomly selecting one rating from each user's rating set, and the rest of the available ratings are considered part of the training set. Weak generalization can assess a model's ability to generalize to other items rated by the same user.
2. *Strong generalization:* It is a two-stage process in which a subset of users is selected randomly and completely removed from the training set to generate the test set. This test set is called **G**. The initial prediction model **M** is trained using all available ratings from the training set (with all the users who are not in the test set **G**). The held-out set is formed by randomly selecting one rating from each user in the set **G** in the second stage (testing phase). During testing, the remaining ratings for each user in set **G** can be utilized to fine-tune the prediction model **M**. The held-out ratings are used to evaluate the model. The primary idea behind strong generalization is to create a prediction model or a large number of initial users that can later be generalized for a small number of new users (**G**).

4.3. Hyperparameter Tuning

Hyperparameters for our neural network architecture were tuned by randomized grid search cross-validation. The optimal hyperparameters for different datasets and different setups are given as follows.

- *MovieLens weak generalization*: In this scenario, the embedding layer sizes for θ, U, and V were set to 32, 200, and 200, respectively. After the embedding layer, the subsequent two layers for θ held sizes 16 and 4. Similarly, the user and item layers shared the

[1]https://grouplens.org/datasets/movielens/1m/
[2]https://grouplens.org/datasets/eachmovie/

following layer sizes: 200, 256, and 512. Dropout regularization with a probability of 0.2 was applied between these layers. Throughout the model, the LeakyReLU activation function was utilized. Furthermore, L2 regularization with a regularization parameter (λ) value of 0.001 is applied to all layer weights.

- *MovieLens strong generalization*: For strong generalization on the MovieLens dataset, we selected 1040 users for our test set **G**, as outlined in section 4.2. Initially, we trained a base model using the training set with the users except for the test set users. In this base model, the embedding layer sizes for θ, user, and item are set to 32, 100, and 100, respectively. A dropout of 0.2 was applied after each embedding layer. The subsequent θ layers had sizes of 16 and 4, with a dropout of 0.2 applied between them. Similar to the weak generalization model, the user and item layers shared layer sizes of 128 and 256, respectively, with a dropout of 0.2 applied between them. The L2 regularization technique with a regularization parameter (λ) value of 0.0015 was employed for all layer weights.

  Next, we trained a new model to predict ratings for a new set of users while maintaining the same collection of items as before. We used the identical structure for the item layers as in the previous model, without any dropout between these layers. We transferred the layer weights learned in the base model to these new item layers and froze them, preventing their further training in this new model. The embedding sizes for θ and the user were set to 32 and 64, respectively, followed by a dropout of 0.2 after both layers. The θ layers held sizes of 16 and 4, with a dropout of 0.2 applied between them. Following the user embedding layer, the user layer had a size of 256 with a dropout of 0.2 applied between its layers. Throughout the model, the LeakyReLU activation function was used. Additionally, L2 regularization with a regularization parameter (λ) value of 0.0015 was applied to all layer weights.

- *EachMovie weak generalization*: In this setting, the embedding layer sizes for θ, U, and V were set to 64, 200, and 200, respectively. After each embedding layer, a dropout layer with a dropout probability of 0.3 is incorporated. The subsequent three θ layers following the embedding layer had sizes 32, 16, and 5, with a dropout of 0.3 between the layer sizes 32 and 16 and a dropout of 0.2 before the final layer of size 5. Similarly, the user and item layers shared 200, 128, and 256 layer sizes. Dropout regularization with a probability of 0.3 is applied between these layers. Throughout the model, the LeakyReLU activation function is used. Furthermore, L2 regularization with a regularization parameter (λ) value of 0.001 was applied to all layer weights.

- *EachMovie strong generalization*: For strong generalization on the EachMovie dataset, we selected 6000 users for our test set **G**, as described in section 4.2. Initially, we trained a base model using the same hyperparameters described in the weak generalization. Subsequently, we trained a new model to predict ratings for a new set of users while maintaining the same set of items as before. We used the same structure as in the previous model for the item layers without any dropout between these layers. We transferred the layer weights learned in the base model to these new item layers and froze them, preventing their further training in this new model. The embedding sizes for θ and the user were set to 32 and 100, respectively, followed by a dropout of 0.3 after both layers. The θ layers held sizes of 16 and 5, with a dropout of 0.2 applied between them. The user layers are of sizes 128 and 256, with a dropout of 0.3 applied between them. Throughout the model, the LeakyReLU activation function was used. Additionally, L2 regularization with a regularization parameter (λ) value of 0.0015 was applied to all layer weights.

We have considered ten well-known algorithms for comparison and these are URP Marlin (2003), Attitude Marlin (2004), Fast MMMF Rennie & Srebro (2005), E-MMMF DeCoste (2006), PMF Mnih & Salakhutdinov (2007), BPMF Salakhutdinov & Mnih (2008), GP-LVM Lawrence & Urtasun (2009), iPMMMF and iBPMMMF Xu et al. (2012) and Gibbs MMMF and iPMMMF Xu et al. (2013), and HMF Kumar et al. (2017).

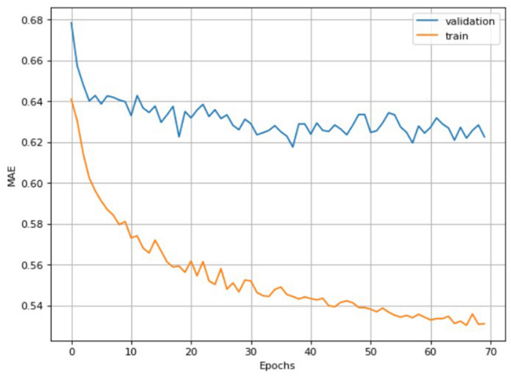
4.4. Evaluation metrics and results

We use Mean Absolute Error(MAE) and Normalized Mean Absolute Error(NMAE) metrics to evaluate the efficacy of the proposed model against the state-of-the-art techniques. The NMAE is defined as MAE divided by a factor that depends on R. The factor is the expected value of the MAE assuming that the rating values and rating predictions are uniformly distributed. For the MovieLens dataset ($R = 5$) the factor is 1.6 and 1.944 for the EachMovie dataset ($R = 6$). The results reported here are an average of 5 runs. you need to specify why 1.6 or 1.944.

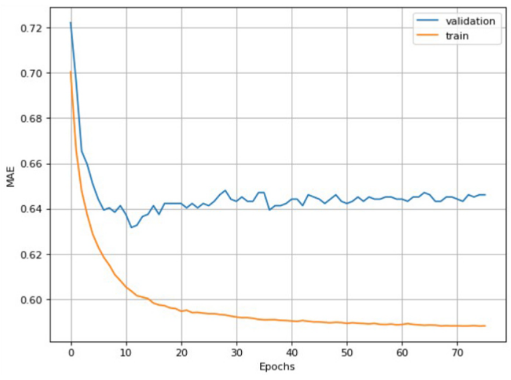
Algorithms	MovieLens		EachMovie	
	Strong	Weak	Strong	Weak
URP	.4341	.4444	.4422	.4557
Attitude	.4320	.4375	.4520	.4550
Fast MMMF	.4156	.4203	.4397	.4341
E-MMMF	.4029	.4071	.4287	.4295
PMF	.4332	.4413	.4466	.4579
BPMF	.4235	.4450	.4352	.4445
GP-LVM	.4026	.3994	.4179	.4134
iPMMMF & iBPMMMF	.4031	.4089	.4211	.4224
Gibbs MMMF & iPMMMF	.4037	.4040	.4134	.4142
HMF	.4019	.4032	.4118	.4095
DeepMMMF	**0.3987**	**0.3954**	**0.4090**	**0.4054**

Table 1: Average NMAE of different models

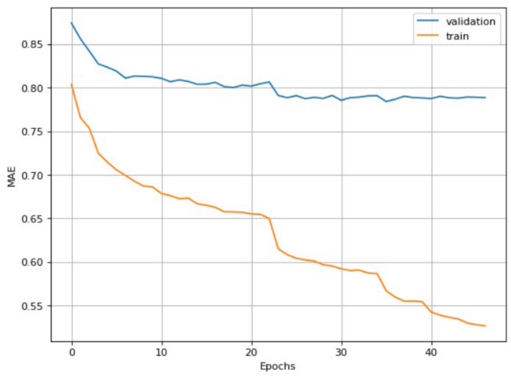
The table compares the proposed approach and various state-of-the-art matrix factorization techniques for collaborative filtering. The analysis evidences that DeepMMMF consistently outperforms the other methods in terms of NMAE in both strong and weak generalizations. In strong generalization, where the testing data contains entirely unseen user-item interactions, DeepMMMF achieves a significantly lower NMAE than the other approaches. It indicates the model's ability to generalize to new instances, effectively capturing complex user-item relationships. Furthermore, even in weak generalization scenarios, where the testing data contains partially seen interactions, DeepMMMF maintains its superiority over the other methods. Its lower NMAE suggests it can accurately predict user preferences, even with limited available data. The results demonstrate the robustness and reliability of DeepMMMF for collaborative filtering tasks. The superior performance of DeepMMMF can be attributed to its deep learning architecture, which enables it to capture intricate patterns and dependencies in user-item interactions. By leveraging the power of deep neural networks, DeepMMMF can effectively model nonlinear relationships, surpassing traditional matrix factorization approaches.



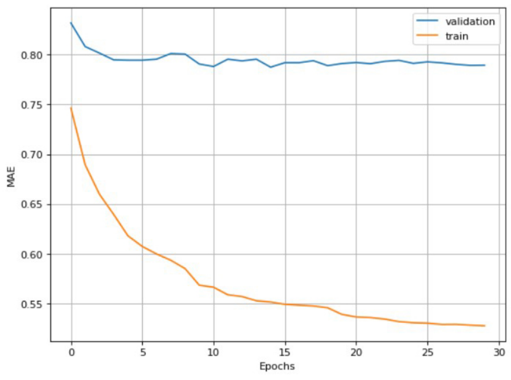
(a) MovieLens Weak Generalization



(b) MovieLens Strong Generalization



(c) EachMovie Weak Generalization



(d) EachMovie Strong Generalization

Figure 2: MAE plots for different models

Figure 2 illustrates the analysis of the proposed DeepMMMF performance in terms of MAE over the number of epochs on different datasets and settings. It is important to note that we used early stopping callback which stops model training when the monitored metric (validation MAE here) stops improving. The reason is that validation MAE might begin to increase with further training. These figures' purpose is to depict the model's performance and progression during training visually. It is essential to monitor the behavior of the validation MAE relative to the training MAE to detect overfitting.

5. Conclusion

This work investigated a deep neural network architecture for matrix factorization with maximum margin separators in the latent space. We demonstrated the advantage of the proposed approach over the baseline approach and provided the empirical analysis to corroborate the claim. The proposed method can be ported into a parallel or distributed environment for faster model training. Incorporating contextual information or location information into the model gives a good scope for the future direction of this research. Further, the proposed approach opens a pathway to investigate kernelized MMMF where one can try different kernel functions on item latent features to model the nonlinear relations between users and the items.

References

Bi, X., Qu, A., Wang, J., & Shen, X. (2017). A group-specific recommender system. *Journal of the American Statistical Association*, *112*, 1344–1353.

Bobadilla, J., Alonso, S., & Hernando, A. (2020). Deep learning architecture for collaborative filtering recommender systems. *Applied Sciences*, *10*, 2441.

Da'u, A., & Salim, N. (2020). Recommendation system based on deep learning methods: a systematic review and new directions. *Artificial Intelligence Review*, *53*, 2709–2748.

DeCoste, D. (2006). Collaborative prediction using ensembles of maximum margin matrix factorizations. In *Proceedings of the 23rd international conference on Machine learning* (pp. 249–256).

Fu, M., Qu, H., Yi, Z., Lu, L., & Liu, Y. (2018). A novel deep learning-based collaborative filtering model for recommendation system. *IEEE transactions on cybernetics*, *49*, 1084–1096.

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T.-S. (2017). Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web* (pp. 173–182).

Koren, Y. (2008). Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* KDD '08 (p. 426–434). New York, NY, USA: Association for Computing Machinery. URL: `https://doi.org/10.1145/1401890.1401944`. doi:`10.1145/1401890.1401944`.

Kumar, V., Pujari, A. K., Sahu, S. K., Kagita, V. R., & Padmanabhan, V. (2017). Collaborative filtering using multiple binary maximum margin matrix factorizations. *Information Sciences*, *380*, 1–11.

Lara-Cabrera, R., González-Prieto, Á., & Ortega, F. (2020). Deep matrix factorization approach for collaborative filtering recommender systems. *Applied Sciences*, *10*, 4926.

Lawrence, N. D., & Urtasun, R. (2009). Non-linear matrix factorization with gaussian processes. In *Proceedings of the 26th annual international conference on machine learning* (pp. 601–608).

Lee, D. D., & Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, *401*, 788–791.

Liu, H., Zheng, C., Li, D., Shen, X., Lin, K., Wang, J., Zhang, Z., Zhang, Z., & Xiong, N. N. (2022). Edmf: Efficient deep matrix factorization with review feature learning for industrial recommender system. *IEEE Transactions on Industrial Informatics*, *18*, 4361–4371. doi:10.1109/TII.2021.3128240.

Marlin, B. (2004). *Collaborative filtering: A machine learning perspective*. University of Toronto Toronto.

Marlin, B. M. (2003). Modeling user rating profiles for collaborative filtering. In S. Thrun, L. Saul, & B. Schölkopf (Eds.), *Advances in Neural Information Processing Systems*. MIT Press volume 16. URL: https://proceedings.neurips.cc/paper_files/paper/2003/file/269d837afada308dd4aeab28ca2d57e4-Paper.pdf.

Mnih, A., & Salakhutdinov, R. R. (2007). Probabilistic matrix factorization. *Advances in neural information processing systems*, *20*.

Paterek, A. (2007). Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD cup and workshop* (pp. 5–8). volume 2007.

Rennie, J. D., & Srebro, N. (2005). Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning* (pp. 713–719).

Salakhutdinov, R., & Mnih, A. (2008). Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th international conference on Machine learning* (pp. 880–887).

Srebro, N., Rennie, J., & Jaakkola, T. (2004). Maximum-margin matrix factorization. *Advances in neural information processing systems*, *17*.

Wu, H., Zhang, Z., Yue, K., Zhang, B., He, J., & Sun, L. (2018). Dual-regularized matrix factorization with deep neural networks for recommender systems. *Knowledge-Based Systems*, *145*, 46–58.

Xu, M., Zhu, J., & Zhang, B. (2012). Nonparametric max-margin matrix factorization for collaborative prediction. *Advances in Neural Information Processing Systems*, *25*.

Xu, M., Zhu, J., & Zhang, B. (2013). Fast max-margin matrix factorization with data augmentation. In *International Conference on Machine Learning* (pp. 978–986). PMLR.

Xue, H.-J., Dai, X., Zhang, J., Huang, S., & Chen, J. (2017). Deep matrix factorization models for recommender systems. In *IJCAI* (pp. 3203–3209). Melbourne, Australia volume 17.

Yi, B., Shen, X., Liu, H., Zhang, Z., Zhang, W., Liu, S., & Xiong, N. (2019). Deep matrix factorization with implicit feedback embedding for recommendation system. *IEEE Transactions on Industrial Informatics*, *15*, 4591–4601.

Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)*, *52*, 1–38.