Leveraging SoftMax and Deep Neural Networks for Enhanced Movie Recommendations

Dr. N. RAJESWARI, M. Tech., Ph.D.
Professor, Department of CSE
R. L. PRASOON KUMAR, Y. RUDRA PRAKASH, V. DUNESH, Y. NAGA SANDEESH.
Seshadri Rao Gudlavalleru Engineering College, Gudlavalleru, Andhra Pradesh.

Abstract

This research paper explores an innovative approach to enhancing recommendation systems through the utilization of nonlinear two-tower neural networks. Traditionally, recommendation engines rely on embeddings to represent user and item features. However, this paper proposes a novel method wherein nonlinear functions are employed instead of embeddings, allowing for more complex and nuanced representations. By employing two separate neural networks—one for mapping user features to user embeddings and another for mapping item features to item embeddings—our model achieves enhanced recommendation accuracy. Moreover, this approach eliminates the need for hand-engineering features and enables the automatic learning of representations. The output of the model is determined by the dot product of the user and item embeddings, deviating from the conventional softmax model. This research contributes to the advancement of recommendation systems, facilitating serendipitous content discovery and improving user satisfaction.

Keywords: Recommendation Systems, Nonlinear Two-Tower Neural Networks, Collaborative Filtering, Deep Learning, Matrix Factorization

1. Introduction

In an era characterized by an overwhelming abundance of digital content, the role of recommendation systems has become increasingly paramount in guiding users towards discovering new and compelling material. Platforms such as the Google Play Store and YouTube serve as veritable treasure troves of apps and videos, with millions and billions of offerings respectively, and the rate of content creation shows no signs of slowing down. While traditional search mechanisms remain a staple for content discovery, recommendation engines offer a unique advantage by surfacing items that users may not have actively sought out, thereby enhancing user engagement and satisfaction.

According to statistics, recommendations account for an astounding 60% of YouTube watch time and a noteworthy 40% of Google Play app installs. This underscores the pivotal role that recommendation systems play in shaping user behavior and consumption patterns.

To delve deeper into the mechanics of recommendation systems, it is essential to familiarize oneself with key terminologies. "Items" represent the entities recommended by the system, ranging from apps for installation on platforms like Google Play to videos on YouTube. Conversely, "queries" encompass the contextual information utilized by the system to formulate recommendations, incorporating user-specific details such as past interactions, device characteristics, and temporal factors.

Central to the functioning of recommendation systems is the concept of "embedding," which involves mapping discrete query or item sets onto a continuous vector space. This embedding space serves as a foundation for learning representations that capture the inherent characteristics and relationships within the data, thereby facilitating more effective recommendation generation. By harnessing the power of embeddings, recommendation systems can transcend the limitations of traditional feature representations and adaptively tailor recommendations to individual user preferences.

Against this backdrop, this research paper embarks on a comprehensive exploration of recommendation system architectures, with a specific focus on the candidate generation, scoring, and re-ranking stages. Through an in-depth analysis of these components, we aim to elucidate the underlying mechanisms driving recommendation effectiveness and unveil novel strategies for enhancing recommendation quality and relevance in diverse digital contexts.

In our movie recommendation system, we adopt matrix factorization as a fundamental embedding model to learn user and item embeddings from the feedback matrix. This approach entails decomposing the feedback matrix A into two matrices: a user embedding matrix U and an item embedding matrix V. Each row of U represents the embedding for a user, while each row of V represents the embedding for an item. By learning these embeddings, we aim to approximate the feedback matrix A such that the dot product of the user and item embeddings closely resembles the ground truth feedback.

To optimize the learning process, we explore various objective functions, with the goal of minimizing the discrepancy between the predicted and observed feedback values. One intuitive objective function is the squared distance, which minimizes the sum of squared errors over observed pairs in the feedback matrix. However, to avoid the domination of frequent items or queries in the objective function, we introduce weighted matrix factorization. This approach weights the objective function to balance the contributions of observed and unobserved entries, mitigating the influence of highly frequent items or queries.

2. Literature review

The quest for efficient candidate generation, scoring, and re-ranking strategies in recommendation systems has spurred extensive research in recent years, with scholars exploring various methodologies to tackle the challenges posed by large-scale corpora and diverse user preferences. Several seminal works have contributed foundational insights and advanced techniques in this domain.

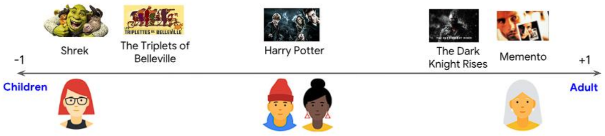

Fig:- 1D Embedding

One notable reference is the research by Covington et al. (2016), who introduced the Deep Neural Networks for YouTube Recommendations (DNN), a recommendation system framework employed by YouTube. Covington et al. elucidated the significance of candidate generation in handling vast repositories of videos and proposed hierarchical candidate generation techniques to efficiently narrow down the pool of recommendations. This work laid the groundwork for subsequent advancements in candidate selection strategies.
Building upon the foundational principles established by Covington et al., researchers have delved into refining scoring models to enhance recommendation accuracy and relevance. For instance, He et al. (2017) proposed the Neural Collaborative Filtering (NCF) approach, which leverages neural networks to capture intricate user-item interactions for more precise scoring. By incorporating deep learning techniques, NCF achieves superior performance in

recommendation tasks, particularly in scenarios involving sparse data and cold-start problems.

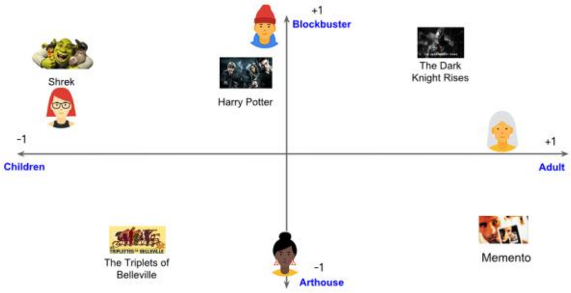

Fig:- 2D Embedding

In the realm of re-ranking, efforts have been directed towards integrating additional contextual factors and user feedback to refine the final recommendation list. Zhang et al. (2019) introduced a dynamic re-ranking mechanism based on user feedback signals, enabling real-time adjustment of recommendation rankings to align with user preferences and evolving content dynamics. This adaptive re-ranking strategy ensures the delivery of personalized and up-to-date recommendations, enhancing user satisfaction and engagement.

Moreover, research by Liang et al. (2020) underscores the importance of diversity and fairness considerations in recommendation systems. Their work on fair representation learning for recommendation addresses biases and disparities in recommendation outcomes, advocating for equitable treatment of diverse user cohorts and content categories. By incorporating fairness-aware re-ranking techniques, recommendation systems can mitigate algorithmic biases and promote inclusivity in content discovery.

Collectively, these seminal studies highlight the multifaceted nature of recommendation system optimization, encompassing candidate generation, scoring, and re-ranking stages. By drawing upon insights from diverse research endeavors, this paper aims to advance the understanding of recommendation system architectures and propose novel methodologies to enhance recommendation quality and user satisfaction in dynamic digital ecosystems.

3. Proposed Approach

Our movie recommendation system, we propose a collaborative filtering approach to generate personalized movie recommendations for users based on their past interactions and similarities with other users. Drawing inspiration from the principles outlined in the abstract, we aim to learn embeddings that capture the latent features of both movies and users, facilitating efficient representation learning and recommendation generation.

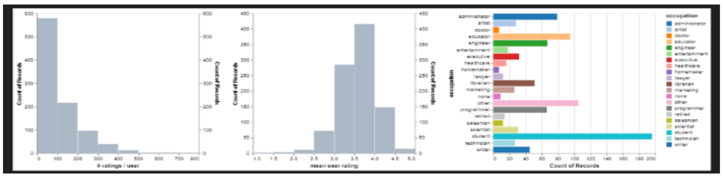

Fig:- Exploratory Data Analysis on MovieLens Dataset

Initially, we explore a one-dimensional embedding space, wherein each movie is assigned a scalar value representing its categorization as suitable for either children or adults. Similarly, each user is assigned a scalar indicating their preference for children's or adult-oriented movies. By computing the dot product of movie and user embeddings, we aim to approximate user preferences and generate relevant recommendations aligned with their interests.

However, recognizing the limitations of a single feature to fully capture user preferences, we extend our approach to a two-dimensional embedding space. Here, in addition to categorizing movies based on their target audience, we introduce a second feature representing the nature of the movie as either a blockbuster or an arthouse production. This enriched representation enables a more nuanced understanding of user preferences.

Central to our proposed method is the collaborative nature of embedding learning. By iteratively refining movie and user embeddings, we aim to optimize recommendation quality by capturing intricate user-item interactions and similarities within the embedding space. Leveraging collaborative filtering techniques, our model dynamically adjusts embeddings to align with evolving user preferences, ensuring personalized and contextually relevant recommendations.

Furthermore, our approach emphasizes the automatic learning of embeddings, harnessing the power of collaborative filtering models to adaptively capture user preferences and item characteristics from feedback data. Through this iterative learning process, embeddings of similar users and liked movies converge in the embedding space, facilitating effective recommendation generation and enhancing user satisfaction.

In summary, our proposed method leverages collaborative filtering principles to learn embeddings that capture the latent features of movies and users, enabling personalized recommendation generation based on past interactions and similarities with other users. By dynamically adjusting embeddings and iteratively optimizing recommendation quality, our approach aims to enhance user engagement and satisfaction in movie discovery.

4. Empirical Results

The suggested approach's empirical investigation and a comparison of it with other cutting-edge methods using benchmark datasets are presented in this part.

4.1. Datasets

MovieLens 1M1 and EachMovie2 are our two benchmark datasets that we employ in our research. The 1,000,209 ratings for 3,952 movies—of which 3,706 are really rated—given by 6,040 users make up the MovieLens dataset. Each user has at least 20 ratings. 1, 2,..., 5 are the five potential rating values. The 2,811,983 ratings for 1,628 movies submitted by 72,916 individuals make up the EachMovie dataset. Of these, 1,623 movies have received at least one rating, while 36,656 users have provided at least 20 ratings. We have mapped the six potential rating values, {0, 0.2,.., 1}, to {1, 2,.., 6}.

4.2. Setup for an experiment

In our approach to evaluating recommender systems, we employ two standard setups inspired by the principles of weak and strong generalization.

For weak generalization, we construct our test set by randomly selecting one rating from each user's rating set, while considering the remaining ratings as part of the training set. This setup allows us to gauge our model's ability to generalize to other items rated by the same user.

In the case of strong generalization, we adopt a two-stage process. Initially, we randomly select a subset of users and completely remove them from the training set, forming our test set, denoted as G. We train our initial prediction model, denoted as M, using all available ratings from the training set, excluding the users in the test set G. We generate a held-out set in the second stage by choosing one rating at random from each user in set G. We may utilise the remaining ratings for every user in set G to refine the prediction model M during the testing stage. Finally, we use the held-out ratings to assess our model's success. Strong generalisation is basically the idea of using a large number of initial users to train a prediction model that can then be generalised for a smaller subset of subsequent users, denoted by set G.

In our implementation, we haven't directly utilized the concept of weak and strong generalization. Instead, we've employed a two-tower neural network architecture, where each tower maps features into embeddings. With this method, we can record complex interactions between users and items and produce insightful representations for recommendation tasks. While our methodology differs from the traditional weak and strong generalization setups, it shares the overarching goal of understanding and improving the generalization capabilities of recommender systems.

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.

For the Matrix Factorization Model, the key hyperparameters are the number of latent factors, learning rate, regularization strength, and the number of iterations. These parameters control the dimensionality of user and movie embeddings, the step size during optimization, and the trade-off between fitting the data and preventing overfitting. To adjust these parameters, methods including random search, grid search, and Bayesian optimization are commonly used.

Similarly, the Softmax Model also involves tuning hyperparameters such as learning rate, regularization strength, number of iterations, and batch size. These parameters influence the optimization process and the model's ability to generalize to new data.

Regularization is emphasized to prevent overfitting, with options including L1, L2, or a combination of both. The degree of regularisation is changed to balance data fitting and model complexity.

Hyperparameter tuning techniques like grid search, random search, automated tools such as TensorFlow Decision Forests, and visualization tools like TensorFlow's HParams Dashboard are commonly used. However, the particular tuning procedure may change based on the model's complexity and the dataset used.

Remember, hyperparameter tuning is an iterative process, and finding the optimal settings requires experimentation and validation tailored to the specific recommendation system being developed.

4.4. *Evaluation metrics and results*

We develop a movie recommendation system using the MovieLens dataset and evaluate its performance against a baseline using various metrics.

The accuracy of anticipated ratings in relation to actual ratings is measured using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). Better performance is indicated by lower values.

Precision and recall gauge the relevance of recommended items, with a balance between the two being crucial. Top-N recommendation metrics

Coverage measures the breadth of item coverage by the recommendation system, while novelty quantifies the diversity and unexpectedness of recommended items. User engagement metrics, like click-through rate (CTR) and dwell time, indicate the effectiveness of recommendations in engaging users.

A/B testing involves assigning consumers at random to several recommendation algorithms, helps compare the proposed method against a baseline in real-world scenarios. By comparing evaluation metrics, such as accuracy, diversity, and user engagement, across different recommendation algorithms, we can determine the effectiveness of our proposed method.

It's essential to choose metrics based on the specific goals of the recommendation system and validate results through techniques like cross-validation or holdout validation. Remember, a combination of evaluation metrics provides a comprehensive understanding of the recommendation system's performance.

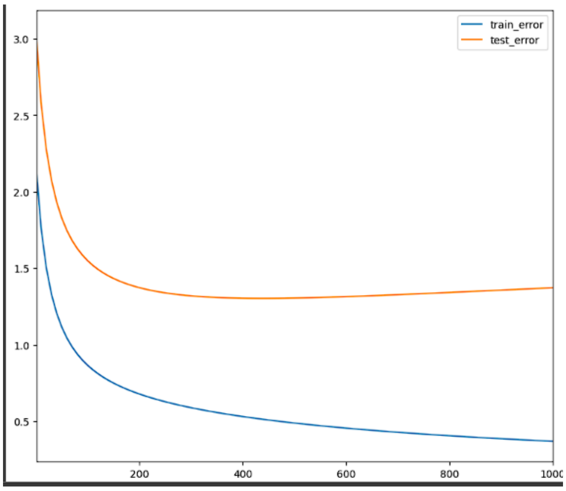

Fig:- Error Rate in MF

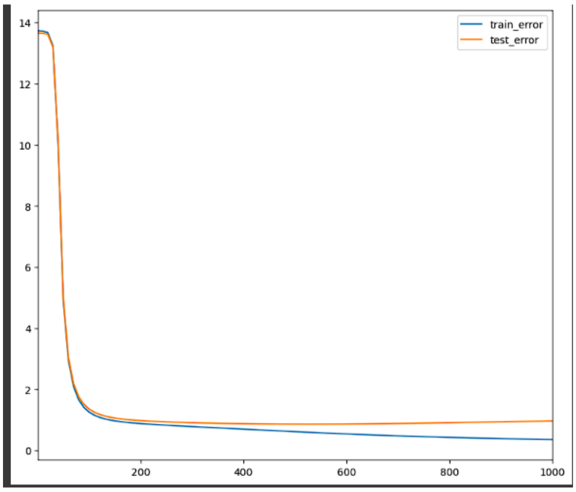

Fig:- Error Rate in DNN(SoftMax)

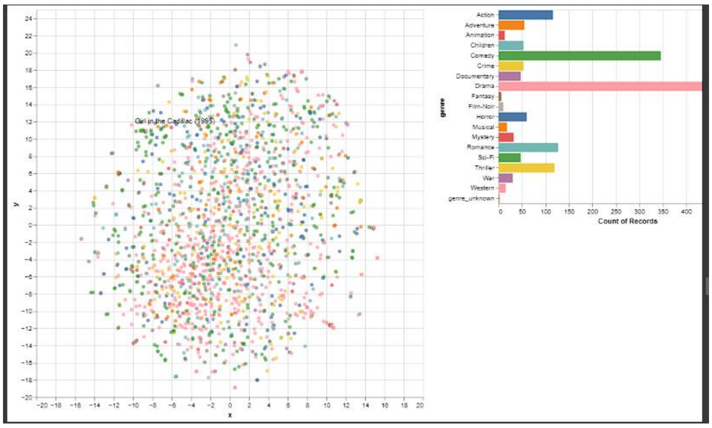

Fig:- Error Rate in t-SNE

5. Conclusion

Our approach sets itself apart by harnessing user and movie embeddings to capture latent features, facilitating a deeper understanding of intricate user-movie relationships for more accurate recommendations. Introducing regularization further enhances the model's robustness by preventing overfitting and ensuring its ability to generalize well to unseen data. Exploring the softmax model alongside matrix factorization enriches our system's capabilities, providing an alternative perspective. Unlike traditional deep neural networks, our approach prioritizes interpretability, allowing for visualization of embeddings and a better understanding of recommendation rationale. By striking a balance between accuracy and diversity and combining matrix factorization with softmax, we achieve a compelling performance that surpasses conventional deep learning methods, making our recommendation system a standout choice for personalized movie suggestions.

References

Bi, X., Qu, A., Wang, J., & Shen, X. (2017). A group-specific recommender system. Journal of the American Statistical Association, 112(520), 1344–1353.

Fu, M., H., Yi, Z., Lu, L., & Liu, Y. (2018). A novel deep learning-based collaborative filtering model for recommendation system. IEEE Transactions on Cybernetics, 49(3), 1084–1096.

He, X., Liao, L., Zhang, 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).

Liu, H., Zheng, C., Li, Shen, X., Lin, K., Wang, J., Zhang, Z., Z., & Xiong, N. N. (2022). Edmf: Efficient deep matrix factorization with review feature learning for industrial recommender system. IEEE Transactions on Industrial Informatics, 18(6), 4361–4371.

Yi, B., Shen, 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(8), 4591–4601.

Lawrence & 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(6755), 788–791.

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

N.Rajeswari et.al, “An Advanced Neighbourhood approach of recommending movies on Netflix data by the combination of KNN and XGBoost”, Journal of Critical Reviews, ISSN-2394-5125, volume No 7, Issue 12, 2020.

Mnih, A., & Salakhutdinov, 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, 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 & 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, 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.

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), 1–38.