An End-to-End Deep Learning Model for Recommender Systems

Abebe Tegene[1,3], Vukosi Marivate[2], Mapundi Banda[2], Abiodun Modupe[1], Valisoa Rakotonarivo[2], and Mathibele Nchabeleng[2]

[1] Department of Computer Science, University of Pretoria, Pretoria, RSA, abebe.tegene@up.ac.za
[2] Department of Mathematics and Applied Mathematics, University of Pretoria, Pretoria, RSA
[3] CoE of Big Data Analytics and HPC, Addis Ababa Science and Technology University, Addis Ababa 16417, Ethiopia

Abstract. The increasing availability of choices on online platforms has resulted in a rise in user expectations for personalized content across social media, entertainment, and e-commerce websites. Recommender systems, using machine learning, analyze user preferences to provide intelligent recommendations, help users manage information overload. Collaborative filtering, particularly based on latent factor models like matrix factorization, has proven successful in personalized recommendations but encounters challenges such as data sparsity and issues with non-linear feature representation. The use of deep neural networks for personalized recommendations has garnered interest to the advancements in deep learning methodologies. To improve recommendation performance and user experience, research is focusing on applying deep learning concepts to address existing challenges. This paper proposes an end-to-end deep learning framework to address these challenges. The fundamental idea of this method involves transforming the dense feature vector produced by embedding methods into two end-to-end deep neural network designs. Subsequently, it independently learns a low-dimensional feature representation and non-linear abstraction of the sparse data. In addition, the method incorporates a deep learning architecture into the output layer of the networks to predict the required rating scores. In four real-world datasets, this proposed technique surpassed state-of-the-art models in terms of performance.

Keywords: deep learning, matrix factorization, collaborative filtering, Recommender systems

1 Introduction

The diversity of choices available to consumers has increased dramatically as a result of the increasing use of the Internet. On modern social media, entertainment, and e-commerce platforms, consumers often find a wide range of well-liked products, movies, music, and restaurants. This results in an increasing volume of

content available on a given platform; thus, users frequently experience information overload, making it challenging to select anything suitable from an extensive number of options. As a result, personalized advice is the core strategy to provide clients with a better user experience [1, 2].

Recommender systems (RSs) have become widely recognized as an essential tool to help consumers navigate a wide range of options while also keeping them interested and happy with personalized content [1, 3]. RSs are classified into three distinct groups, based on the input data they use: collaborative filtering (CF), content-based (CB), and hybrid [4]. CF is an effective implementation strategy that makes recommendations based on previous user-item interactions, using an explicit or implicit interaction data matrix; however, it suffers from different issues: data sparsity, with non-linear feature representation, and cold start problems [5, 6]

Recent advances in deep learning (DL) have produced state-of-the-art results in a variety of fields, including image recognition, natural language processing, computer vision, RSs, and many more [7, 8]. The DL method can accomplish automatic feature extraction, while traditional RSs techniques require human feature extraction [9]. Because there are more users and items than ever before, the RS issues have high-dimension data. Thus, DL can be used to automatically extract high-dimensional rich feature representations from these data [10, 11]. Even though DL has had some successful results in other domains. Those issues are still exist in RSs [7]. Therefore, it has become essential to find a more robust method of applying DL techniques in the field of RSs to address the aforementioned problems and improve their performance.

In this work, we explore the benefits of the DL framework and introduce a DL method to enhance the performance of RSs. We introduce an end-to-end DL method for these systems, referred to as EDRS. To enhance the effectiveness of recommendations, EDRS uses embedding and DNN (Deep Neural Network) structures to acquire the hidden latent features. The model combines deep learning techniques to extract complex abstractions and nonlinear feature representations from the data for predicting ratings. The core idea is to transform the dense user and item vectors generated by the embedding methods into two fully connected DNN architectures.

In general, in most existing RS designs that utilize DL, user and item input vectors are combined and processed to pass through a shared DNN to extract features. We believe that this approach lacks the resilience needed to effectively capture the necessary information from the data. Due to the intricate nature of user behaviors, we recommend analyzing these aspects separately to acquire the relevant insights. We contend that the current strategy falls to robustly learn the essential features from the data. Previous methods have not differentiated the architectures for these distinct components. In contrast, the proposed technique assigns the input vectors of the user and the item to its fully connected deep neural networks. Additionally, the method integrate a multi-layer perceptron (MLP) architecture into the proposed method to predict the desired rating scores. This model architecture effectively captures the intricate features

of user and item within their respective frameworks, suggesting that utilizing this proposed end-to-end architecture will significantly improve the performance of recommendations.

To summarize, the key contributions of the proposed work are as follows:

- We developed a DL architecture to factorize a user-item interaction matrix. This approach involves learning the non-linear and the abstract hidden features of users and items within user and item deep learning structures.
- We developed an end-to-end DL framework for recommender systems that effectively addresses the sparsity issues.
- We incorporated an MLP network into the output layer of the proposed architectures to predict rating scores. This integration demonstrates that the techniques significantly enhance recommendation quality, achieving state-of-the-art performance levels.

2 Related Work

This section reveals pertinent research studies about the current study of recommender systems.

2.1 Collaborative Filtering

The CF-based recommendation method works under the fundamental premise that users would select products that are similar to themselves if they had similar interests. Thus, identifying users who share the target user's interests and preferences is crucial for CF-based RSs [12, 13]. For instance, probabilistic matrix factorization (PMF) is one of the CF approaches the model user-item rating matrix as a product of two low-dimensional latent factor matrices using Probabilistic framework to predict missing rating. This method overcomes sparsity problems [14].

One of the efficient latent factor techniques in the CF method is the matrix factorization (MF) model. The method decomposes a high-dimensional data matrix into the product of two new low-rank matrices. This technique maps the latent feature components towards a common latent space. Next, user preferences for items in this space are predicted using the dot product within the latent feature vectors of the user and item [15, 16, 17].

$$\sum_{i,j} e_{ij}^2 = \min_{p_i, q_j} \sum (r_{ij} - \hat{r}_{ij})^2. \tag{1}$$

In most MF models, the latent factors p_i and q_j are typically calculated using stochastic gradient descent methods to optimize the loss given in (Eqn. 1) [17, 18, 19]. In this regard, SVD [17, 18], is one of the best MF techniques available to improve scalability and reliability problems. For example, biased SVD included biasing characteristics in the model to enhance rating prediction [18]. The SVD++ method produced good prediction performance by breaking down

the rating interaction matrix into low-rank matrices [19]. In addition, Regularized Matrix Factorization (RMF) overcomes overfitting and improves generalization in RS tasks by assigning regularization terms to the users and item latent factors [20].

All of these studies demonstrate the value of the MF technique as a foundational tool for RSs and highlight its numerous benefits for mixing different types of data. However, these approaches lack robustness, since they need too many iterations to find appropriate latent characteristic representations. To avoid this, Rendle, Steffen [21] recommended utilizing a large dimension. However, this also affects the quality of the recommendation in the event of a sparse data matrix. Therefore, the proposed approach is well suited to solve the problem.

2.2 Deep Learning Methods

Recent developments in DL-based recommendations have received great interest due to their significant ability to surpass the limitations of conventional methods and generate exceptionally high-quality recommendations [7, 22]. These capabilities enable DL-based RSs to improve user satisfaction

To generate an end-to-end model, the DeepFM [23] combines the factorization machine (FM) and MLP. This system's DNN captures the data's abstract representation. The FM employs two operations, addition and inner product to capture pairwise and linear interactions between features. Mongia et al. proposed a deep latent factor model(DLFCF) that uses multiple nonlinear layers to capture the complex user-item interactions. The method showed better performance over traditional MF methods [24]. Xue, Hong-Jian, et al also developed a deep matrix factorization (DMF) model that extends traditional MF models to learn nonlinear interaction of users and items latent factors using DNNs,and showed astonishing performances [15].

Alashkar, Taleb, et al. proposed a cosmetics recommendation model using MLP [25]. The method employs expert rules and labels instances using two identical MLPs. Using MLP and MF approaches, the NN architecture can be used within the inner product framework. For example, in NNMF [26], a feedforward neural network takes on the role of the classic inner product. In Neural Collaborative Filtering (NCF) framework, He, Xiangnan, et al. integrate the output of an MLP coupled with the latent elements of MF using a nonlinear transformation, before making the final recommendation, [11].

Abebe et al. proposed a deep learning-based matrix factorization method by extending traditional MF using DNNs. The model captures complex and high-order user and item interactions, and improved the accuracy of the recommendation much better than MF models [27, 28]. Zheng et al. proposed a convolution neural network (CNN) to learn the latent representation of users and items. By capturing local patterns, the method improves the accuracy of the rating prediction [29]. Adaptive deep RS (ADRAS) further adapts to both rating and ranking prediction tasks, showing improved performance due to its adaptive strategy [30].

As illustrated, the fundamental issues with the RS tasks are: a significant amount of data is shared between users and items; sparsity; and handling non-linear interaction problems. These results make feature learning a challenging task. Therefore, to address the drawbacks, this paper proposes an end-to-end deep learning method to discover the latent factors. It also integrates embedding approaches to address sparsity issues in the user-item interaction data matrix. In conclusion, enhancing the performance of recommender system models through DL methods is the goal of this paper.

3 Methodology

3.1 Problem Definition

Let $P = \{p_1, p_2, p_3, \ldots, p_m\}$ and $Q = \{q_1, q_2, q_3, \ldots, q_n\}$ represents the sets of m users and n items in the matrix, respectively. Let $Y \in \mathbf{R}^{m \times n}$ be the rating matrix, where $s_{ij} \in Y$ denote the rating given by user i to item j. The user-item interaction rating matrix I is then constructed, using (Eqn. 2).

$$r_{ij} = \begin{cases} s_{ij} & \text{if } s_{ij} \in Y. \\ unknown & otherwise \end{cases} \tag{2}$$

3.2 General Architecture EDSRS

The ultimate goal of the proposed method is to investigate whether DL-based prediction is superior to that of simple dot products. To this end, an end-to-end DL architecture is proposed and depicted in Fig. 1.

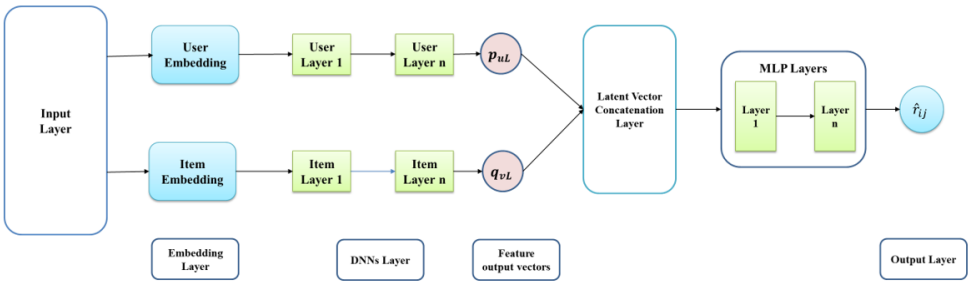


Fig. 1: The architecture of EDRS.

The method has two separate DNN architectures for user and item latent feature representation leaning. The approach integrates a dual DL and embedding techniques to extract abstract and non-linear characteristic representations

from the interaction information for the rating prediction task. That is, the latent factors from the two deep neural network architectures are merged and transferred into an MLP layer to jointly extract and predict the rating scores. That is, to solve the problems associated with latent factor models, such as sparsity, as well as the capacity to elicit deep abstract non-linear characteristics from data, the proposed method employs an end-to-end DL framework in the proposed strategy that incorporates embedding techniques to tackle those difficulties. The basic premise of the approach is mapping the dense latent vectors produced by embedding methods into two fully integrated user- and item-DNN structures. The rating value is then estimated using the DL method called MLP in the output layer of the network between the results of the two latent factors generated by the two structures. The details of the proposed architecture are discussed as follows:

Input layer: The proposed model's input layer takes explicit feature vectors that are derived from user interactions with items.

Embedding Layer: The EDRS model's embedding layer creates a dense, low-dimensional representation from high-dimensional input feature and sparse data. The basic feature vectors for user p and item q can be obtained in this embedding space in the manner described below:

Let for k embedding dimension, $P \in \mathbf{R}^{m \times k}$ be user embedding matrix and $Q \in \mathbf{R}^{n \times k}$ be the item embedding matrix that captures the latent features of users and items, respectively. The proposed method factorizes the rating matrix R into two low-rank matrices that satisfy $R \approx PQ^T$: using the loss function L in Eqn. (3) from known interaction ratings.

$$L = \min_{P,Q} \sum_{(i,j) \in R_{known}} (r_{ij} - p_i {q_j}^T)^2. \tag{3}$$

DNN Layers: This approach involves separately assigning the dense latent factor of user and the dense latent latent factor of item to two DL frameworks, allowing the model to capture nonlinear patterns and intrinsic abstractions within the dataset. In the user architecture of the proposed method, Eqn. (4) is used. In the equation $w_{u1}, w_{u2}, \ldots, w_{uL}$ and $b_{u1}, b_{u2}, \ldots, b_{uL}$ represent user weights and biases, where as σ in this context denotes an activation function, which in our case was $ReLu$.

$$\begin{aligned} h_{u1} &= \sigma(w_{u1}^T h_{u0} + b_{u1}), \\ h_{u2} &= \sigma(w_{u2}^T h_{u1} + b_{u2}), \\ &\vdots \\ h_{uL} &= \sigma(w_{uL}^T h_{uL-1} + b_{uL}), \end{aligned} \tag{4}$$

From each layer, we have an output $h_{u1}, h_{u2}, \ldots, h_{uL}$. Similarly, we obtained $s_{u1}, s_{u2}, \ldots, s_{uL}$ for the output layers of the items.

3.3 Problem Formulation for Prediction Layer

The proposed method utilizes a mapping $\phi : \mathbf{R}^k \times \mathbf{R}^k \rightarrow \mathbf{R}$ that combine two k-dimensional vectors $p \in \mathbf{R}^k$ and $q \in \mathbf{R}^k$ into a single score. For instance, if p denote the latent factor of a user u and q denotes the latent factor of an item v, then $\phi(p, q)$ represents the affinity of the user u to an item v.

In the proposed method, the h_{uL} and s_{uL} are the output features of a dual deep neural network architecture. This is used to predict the rating scores. Then, the dot product between the two factors can be defined using ϕ_{dot}:

$$\phi_{dot} = \langle h_{uL}, s_{uL} \rangle = h_{uL} s_{uL}^T \tag{5}$$

According to [31], MLPs are recognized as universal approximators that can estimate any continuous function in a compact set, provided that they possess an adequate number of hidden states. The following shows how a one layer MLP works:
A one layer MLP function $f : \mathbf{R}^s \rightarrow \mathbf{R}^t$ is defined as:

$$f_{W,b}(x) = \sigma(Wx + b) \tag{6}$$

where $W \in \mathbf{R}^{s \times t}$ and $b \in \mathbf{R}^t$ are parameters and σ is an activation function. MLP is a device that stacks many layers of function f as a composition for approximations. For example, a four layer MLP would have this structure,

$$f_{W_4,b_4}(f_{W_3,b_3})(f_{W_2,b_2}(f_{W_1,b_1}(x)) \tag{7}$$

Consequently, the proposed method merged the two latent factors as $\langle h_{uL}, s_{uL} \rangle$ and applied an MLP function to predict the rating scores as follows:

$$\phi_{MLP} = f_{W_n,b_n}(\dots, (f_{W_1,b_1}([h_{uL}, s_{uL}], \dots) \tag{8}$$

Nowadays, it has become popular to replace the dot product with DNN architectures. Most frequently, several scholars used MLP for their proposed architecture [11, 26, 32]. The reasoning behind this is that since MLPs are universal function approximators, they rigorously outperform fixed similarity functions like the inner product. This indicates that the proposed approach supports the hypothesis of the researchers.

Once the latent features h_{uL} and s_{uL} for users and items are acquired from the two DNN architectures, either of the following two equations is used to predict the target rating score, $\hat{r}_{ij}$.

$$\hat{r}_{ij} = \phi_{MLP} = f_{W_n,b_n}(\dots, (f_{W_1,b_1}([h_{uL}, s_{uL}], \dots) \tag{9}$$

$$\hat{r}_{ij} = \phi_{dot} = h_{uL} \odot s_{uL}, \tag{10}$$

where $\odot$ represents the inner product.

A crucial element in optimizing a recommendation model is having a well-designed objective function. We employ the mean squared error loss function to

optimize model parameters, as it has shown excellent performance with explicit point-wise data types [15].

$$L = \min_{p,q} \sum_{i,j} (r_{ij} - \hat{r}_{ij})^2 + \lambda W^2 + \alpha b^2. \tag{11}$$

where λ and α are regularization parameters for the weights and biases, respectively.

4 Experiments

4.1 Experimental Datasets

This research uses openly accessible MovieLens[4] datasets provided by the GroupLens research team. These datasets consist of movie ratings of different sizes and are widely used to evaluate CF algorithms [11]. For comparing models, we selected a dataset with varients of $100K$ and $1M$. In addition, the paper used an Amazon dataset. The Amazon dataset[5] is the largest RS dataset that has been widely used in various related works [30] (see Table 1).

Table 1: MovieLens and Amazon datasets.

Datasets	**No. of users**	**No. of tems**	**No. of ratings**	**Sparsity**
ML-100K	944	1683	100,000	93.7%
ML-1M	6040	3706	1,000,208	95.5%
Amazon Instant Videos	5100	1596	37,132	99.5%
Amazon Musical Instruments	1498	1022	11,206	99.3%

4.2 Evaluation protocol

To ensure the correctness of the recommendation result, the models' estimating performance is examined using MAE and RMSE [11].

4.3 Baseline Methods

The proposed model is evaluated with the following baselines [24]

– SVD++ is a variation on biased SVD. It enhances the biased SVD model by including implicit information. Implicit feedback, such as surfing and purchasing history, might suggest user preferences when explicit feedback is unavailable. It produced a better recommendation results than regular SVD or biased SVD [33].

[4] https://files.grouplens.org/datasets/movielens/
[5] http://jmcauley.ucsd.edu/data/amazon

- PMF is a MF method that incorporates probabilities. It uses MF and probabilistic models to generate predictions based on given data, It provides a more robust and accurate model for a variety of applications [14].
- NCF:The NCF paradigm improves the recommendation process by using neural networks. It captures intricate user preferences and item information, allowing it to provide personalized recommendations [11].
- DLFCF is a latent factor-based CF model. To find their latent representation vectors, the model applies deep factorization to both users and items [24].
- DMF is a Deep MF model that leverages MLP to transform the user and item representation [15].
- RMF: The model minimizes the distances within the matrices by using the Manhattan distance. The issue of data sparsity is effectively resolved [20].
- DCF is a MF technique for CF that is based on DL. To learn latent features of user and item in their respective structures, it created a dual DNN architecture [27].
- CoNN Deep: [29] This method used CNNs to learn latent factor representations of the user and the items for the rating prediction task.
- ADRS model [30] is an adaptive DL-based model that uses DL techniques to address rating and ranking prediction tasks in RSs.
- DELCR is a DL and embedding-based method for CF. It uses a dot product in the output layer of the network to predict the rating score [28].

4.4 Parameter Settings

Training data makes up 70% of the data, while test and validation data make up the remaining portion. A representative prediction result is obtained by selecting the average value from five distinct training runs, as indicated in Table 2. Tensorflow[6] is used in simulating the model.

For model optimization, Adam [34] is used with a batch size of 512, a learning rate of 0.0001, and a regularization coefficient λ of $1e^{-6}$. For dual DNN, the technique uses two hidden layers. The number of factors in hidden layers is $80, 40$ for the user and for the items, while the number of embedding dimensions is set to 10. In the final layer of the MLP, two hidden layers are used with 80 and 40 neurons, respectively, along with a $ReLu$ activation function in the output layers.

4.5 Experimental Results

The performance result and the percentage improvement result of the proposed method, are summarized in Table 2 and Table 3, respectively. The bold result is achieved using the proposed method. From the result, it can be inferred that the method gives a significantly improved overall performance result for all baseline methods.

[6] https://www.tensorflow.org

Table 2: Model performance result for ML-100K and ML-1M dataset.

Methods	ML-100K		ML-1M	
	MAE	**RMSE**	**MAE**	**RMSE**
SVD++	0.736	0.936	0.703	0.888
PMF	0.728	0.920	0.695	0.873
NCF	0.755	0.961	0.695	0.873
RMF	0.732	0.938	0.689	0.876
DMF	0.735	0.940	0.691	0.878
DLFCF	0.717	0.901	0.678	0.854
EDRS	**0.626**	**0.831**	**0.605**	**0.801**

Table 3: The Percentage improvement result in EDRS for both datasets.

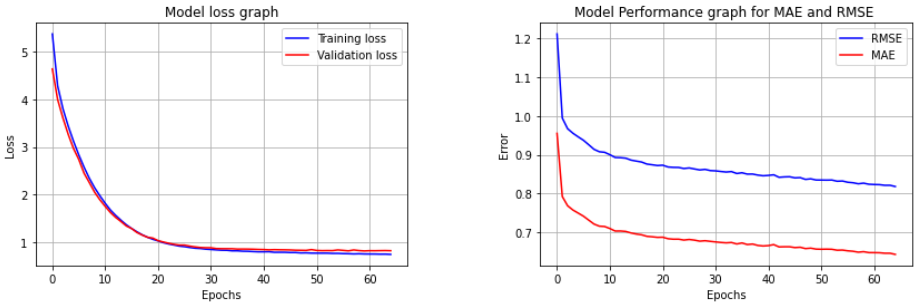
Methods	ML-100K		ML-1M	
	MAE	**RMSE**	**MAE**	**RMSE**
SVD++	14.95%	11.22%	13.94%	9.80%
PMF	14.01%	9.67%	12.95%	8.25%
NCF	17.09%	13.53%	12.95%	8.25%
RMF	14.48%	11.41%	12.19%	8.56%
DMF	14.83%	11.60%	12.45%	8.77%
DLFCF	12.69%	7.77%	10.77%	6.21%

5 Discussion

5.1 Performance Comparison

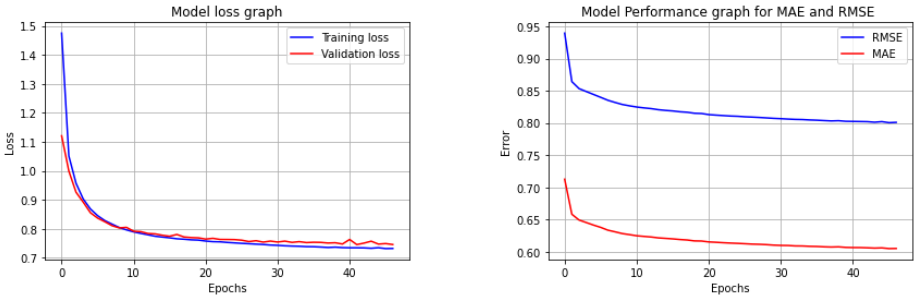
The figures referenced, namely Fig. 2a, and 3a, demonstrate that the suggested model achieves convergence across all datasets. In summary, here are the key points of the experiments:

As shown in Table 3, the proposed method, EDRS, surpasses all other approaches in all datasets. Specifically, compared to the highly regarded DMF model, the EDRS method achieves a relative enhancement of 14.30% in the mean absolute error and 11.60% in RMSE for the variation of 100K dataset. Moreover, it demonstrates an improvement of 12.45% in MAE and 8.77% in RMSE for the 1M dataset. Furthermore, compared to other DL-based models, the advances achieved by the proposed model are substantial. Similarly, the proposed method outperforms the NCF model, which utilizes an MLP in the output layer by a significant margin. This improved performance is due: (i) the utilization of embedded ratings, which effectively addresses the issue of sparse dataset; (ii) separate mapping of the dense embedded latent features to the proposed architecture aids in effectively uncovering the complex and abstract feature representation of users and items; and (iii) incorporation of an MLP layer in the output layer leads to a strong recommendation performance. The difference between EDRS and MF-based methods is once again quite significant. Compared to the 100K dataset, the EDRS method showed a 14.01% improvement in MAE and an 11.41% improvement in RMSE over the RMF method. Similarly, for



(a) Model's loss curve (b) Model's performance curve

Fig. 2: The MAE and RMSE results of EDRS for MovieLens 100K dataset.



(a) Model's loss cruve (b) Model's performance curve

Fig. 3: The MAE and RMSE results of EDRS for MovieLens 1M dataset.

the ML-1M dataset, there was a an improvement of 12.19% in MAE and 8.56% in RMSE. The proposed technique outperforms the other MF methods, such as SVD++, by a considerable margin when compared to robust matrix factorization (RMF) methods. This demonstrates that utilizing an end-to-end deep learning architecture enables the method to effectively capture the underlying feature representation in a sparse dataset, which was a significant limitation of the traditional MF-based models.

The EDRS method, also evaluated based on the latent factor models DLFCF and DCF, shows relatively minimal improvement compared to the other methods. This is not surprising given that the model used a latent factor strategy. However, the improvement seen in the proposed model is attributed to its effective architecture to extract latent features in a robust manner. This shows that the proposed method, EDRS addresses the issue of data sparsity. Overall, EDRS performs well in terms of improving recommendation performance in all datasets. The improvement in performance underscores the effectiveness of the architecture in the method.

Table 4 shows the performance of the EDRS method with respect to the Amazon dataset. As shown, the EDRS model outperforms all baseline approaches, including the Deep-CoNN method. The improvements achieved by EDRS over baselines are significant. This performance improvement shows the effectiveness of the EDRS model compared to existing DL-based recommendation methods.

Table 4: The performance result of EDRS for the Musical Instrument and Instant Video Amazon dataset.

Methods	**Musical Instruments**		**Instant Videos**	
	MAE	RMSE	MAE	RMSE
PMF	0.982	1.004	0.907	1.101
Deep-CON	0.786	0.806	0.691	0.878
ADRS	0.525	0.766	0.505	0.757
EDRS	**0.515**	**0.751**	**0.501**	**0.696**

By evaluating the proposed models on these diverse datasets, spanning the e-commerce, review, and movie domains, we were able to gain a more comprehensive understanding of the models' performance characteristics and their ability to generalize across different contexts. The results of the expanded evaluation showed that the models maintained strong performance in the various datasets, demonstrating their robustness and adaptability to different data characteristics and use cases. This provides greater confidence in the effectiveness of the approaches and their potential for practical deployment in real-world scenarios. Overall, the expanded evaluation has strengthened the findings of the chapter and provided valuable insights into the generalizability and practical applicability of the proposed recommender system models.

5.2 Impact of Dual Deep Learning Layer

The proposed EDRS model is a novel deep learning architecture that uses an end-to-end dual deep learning architecture, which is a significant technical contribution to the field of recommender systems. To expand on this innovation, we provide a comprehensive overview of model architecture and the rationale behind the dual-layer design. When comparing EDRS with one of the representative deep learning-based baseline NCF model both utilize a Multi-Layer Perceptron (MLP) in the output layer, but as observed in Fig. 4 EDRS leverages a Dual Deep Learning Layer, which enables it to significantly outperform the NCF baseline.

An important distinction between EDRS and DELCR is the output layer design. DELCR uses an inner product operation, which likely enables more direct modeling of feature inter-actions, compared to the generic MLP used in EDRS and NCF. The consistently superior performance of both the EDRS and DELCR models, achieved through the innovative dual Deep Learning Layer architecture,

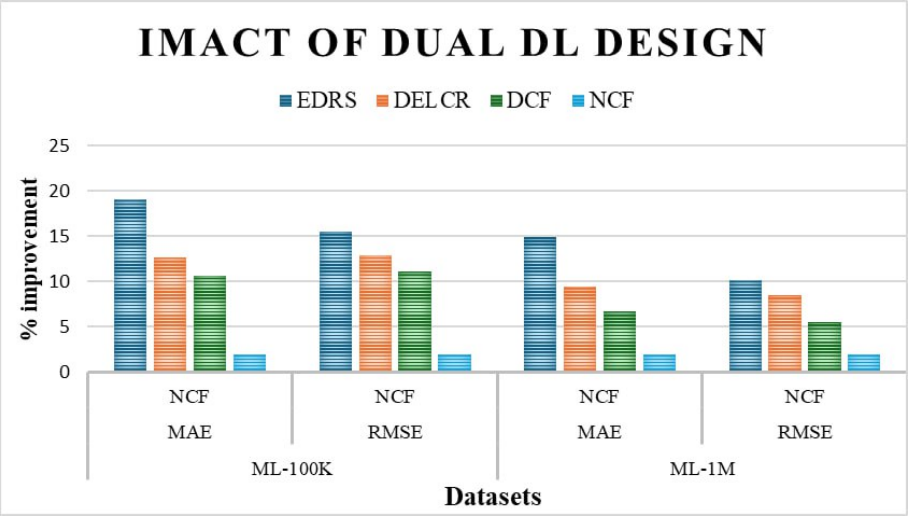


Fig. 4: The architecture advancement impact of the proposed model EDRS.

Table 5: The Performance result of EDRS based on MLP over dot product for ML-100K and ML-1M datasets.

Models	ML-100K		ML-1M	
	MAE (%)	RMSE (%)	MAE (%)	RMSE (%)
DCF	0.698 (+10.32)	0.887 (+6.31)	0.662 (+8.61)	0.842 (+4.87)
DELCR	0.674 (+7.12)	0.856 (+2.92)	0.643 (+5.91)	0.818 (+2.08)

highlights the value of exploring complex deep learning designs to unlock better performing models, which is a significant contribution to the field of deep learning-based recommender systems.

From this we can infer that the innovation of the dual deep learning layer architecture is a significant contribution of the proposed model to the field of deep learning-based recommender systems. This architectural design, which adds an extra deep learning layer compared to baseline models, is a core technical advancement that enables the model to learn more sophisticated representations and achieve substantial performance improvements over existing approaches. The emphasis on the dual deep learning layer as the primary innovation and technical contribution of the proposed model highlights how advancements in deep learning architecture design can drive meaningful progress in developing more effective recommender systems.

5.3 Dot Product vs MLP

The primary goal of this section is to determine whether MLP-learned similarity is a better option than a a simple dot product. As can be seen in Table 5, an

MLP based model substantially outperforms the dot product based baselines on all datasets. For instance, the relative improvement achieved by EDRS using MAE over DCF and DELCR for the ML-100K dataset is 10.32% and 7.12%, respectively. This result demonstrated that EDRS achieved the best performance in improving recommendation performance. The experimental results validate the claim in [11] that a dot product model's embeddings through an MLP can improve the model. The findings of the variants of NeuMF [11] further support the proposition that a learned similarity via an MLP is better than a dot product. In conclusion, the findings confirm that a learned resemblance with an MLP is better than a dot product.

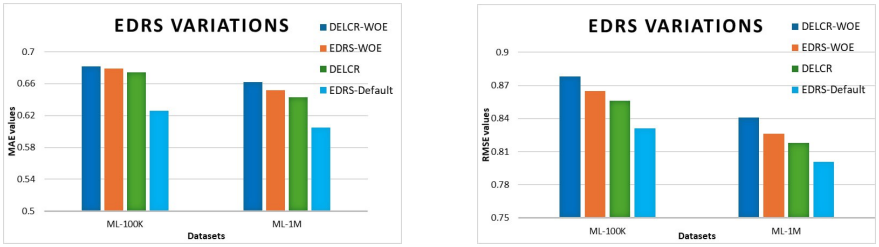
5.4 Ablation Study

To explore the impact of different elements of the EDRS model, this section presented an ablation study by testing the model in various configurations. Figure 5 shows the ablation results.

EDRS-Default represents the baseline configuration of the method presented in this paper. This variation is specifically designed to assess the impact of incorporating DL in the network output layer, achieved by replacing the inner product with a multilayer perceptron.

DELCR : This variation is specifically designed to assess the impact of incorporating dual DL in the network and the use of the inner product in the output layer of the architecture.

DELCR-WOE (without Embedding): This variant was created to evaluate the influence of the embedding techniques. In this configuration, the embedding network is excluded, and the sparse user-item interaction matrix is filled directly with zero values. EDRS-WOE (without Embedding): This version is specifically



(a) Ablation result for MAE

(b) Ablation result for RMSE

Fig. 5: The Ablation result of the proposed mdel using MovieLens dataset.

tailored to assess the impact of embedding techniques in EDRS. In this setup, the embedding network is omitted, and the sparse user-item interaction matrix is directly populated with zero values.

As depicted in Fig. 5, the different configurations of the EDRS method demonstrate varying performance levels in diverse datasets. The EDRS-default,

which represents the standard configuration of the proposed approach, generally outperforms other setups such as DELCR-WOE, EDRS-WOE and DELCR. This emphasizes the substantial influence of end-to-end deep learning strategies on performance improvement.

The EDRS-default shows superior performance, emphasizing the crucial role of embedding techniques and deep learning methods in boosting network performance. In contrast, the EDRS-WOE setup, which omits the embedding network and directly incorporates zero values into the sparse user-item interaction matrix, still achieves good results. This demonstrates the effectiveness of the multilayer perceptron in improving model performance and underscores the beneficial influence of embedding strategies on the accuracy of recommendation systems.

Furthermore, the DELCR version demonstrates relatively strong performance because it utilizes both dual deep learning and embedding techniques within the network. This highlights the importance of these approaches for enhancing the performance of recommendation systems.

Moreover, the DELCR-WOE version shows low performance due to its sparse user-item interaction matrix resulting from the direct insertion of zero values into the sparse user-item interaction matrix. This indicates that neglecting embedding approaches reduces model accuracy, emphasizing the importance of these strategies for improved recommendation system performance.

6 Conclusion and Future Works

In this study, we investigated an end-to-end deep learning approach for recommender systems. The method involves initially embedding the input vectors of users and items to create dense low-dimensional representations that are then individually fed into two DNN architectures to unveil abstract and nonlinear data representations. Subsequently, the outputs from these structures are combined and processed using an MLP layer to predict ratings. This framework effectively addresses issues seen in traditional matrix factorization models, yielding notable performance enhancements and outperforming existing DL-based RS methods. Through extensive experiments on real-world datasets, the proposed model demonstrated superior performance in rating prediction compared to state-of-the-art techniques. These results suggest that DL methodologies are crucial for enhancing recommendation system performance. Our future direction includes integrating additional deep learning frameworks into recommendation systems to proactively boost recommendation quality. In addition, we plan to incorporate large language models into the recommender system domains.

Acknowledgement. The authors gratefully acknowledge the support of the ABSA Chair of Data Science and the Data Science for Social Impact (DSFSI) Lab at the University of Pretoria. This work was supported by UK International Development and the International Development Research Centre (IDRC), Ottawa, Canada, under the AI4D Africa Program. DSFSI also acknowledges gifts from NVIDIA, Google.org, OpenAI, and Meta.

References

[1] Brent Smith and Greg Linden. "Two decades of recommender systems at Amazon. com". In: *Ieee internet computing* 21.3 (2017), pp. 12–18.

[2] Yong Zheng and David Xuejun Wang. "A survey of recommender systems with multi-objective optimization". In: *Neurocomputing* 474 (2022), pp. 141–153.

[3] Sen Li et al. "Embedding-based product retrieval in taobao search". In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2021, pp. 3181–3189.

[4] Aminu Da'u and Naomie Salim. "Recommendation system based on deep learning methods: a systematic review and new directions". In: *Artificial Intelligence Review* 53.4 (2020), pp. 2709–2748.

[5] Francesco Ricci, Lior Rokach, and Bracha Shapira. "Introduction to recommender systems handbook". In: *Recommender systems handbook*. Springer, 2010, pp. 1–35.

[6] Yehuda Koren. "Factor in the neighbors: Scalable and accurate collaborative filtering". In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 4.1 (2010), pp. 1–24.

[7] Shuai Zhang et al. "Deep learning based recommender system: A survey and new perspectives". In: *ACM computing surveys (CSUR)* 52.1 (2019), pp. 1–38.

[8] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *nature* 521.7553 (2015), pp. 436–444.

[9] Ayush Singhal, Pradeep Sinha, and Rakesh Pant. "Use of deep learning in modern recommendation system: A summary of recent works". In: *arXiv preprint arXiv:1712.07525* (2017).

[10] Yao Wu et al. "Collaborative denoising auto-encoders for top-n recommender systems". In: *Proceedings of the ninth ACM international conference on web search and data mining*. 2016, pp. 153–162.

[11] Xiangnan He et al. "Neural collaborative filtering". In: *Proceedings of the 26th international conference on world wide web*. 2017, pp. 173–182.

[12] Xiaoyuan Su and Taghi M Khoshgoftaar. "A survey of collaborative filtering techniques". In: *Advances in artificial intelligence* 2009 (2009).

[13] Mohammed Fadhel Aljunid et al. "A collaborative filtering recommender systems: Survey". In: *Neurocomputing* 617 (2025), p. 128718.

[14] Andriy Mnih and Russ R Salakhutdinov. "Probabilistic matrix factorization". In: *Advances in neural information processing systems* 20 (2007).

[15] Hong-Jian Xue et al. "Deep matrix factorization models for recommender systems." In: *IJCAI*. Vol. 17. Melbourne, Australia. 2017, pp. 3203–3209.

[16] Xiangnan He et al. "Fast matrix factorization for online recommendation with implicit feedback". In: *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 2016, pp. 549–558.

[17] Yehuda Koren, Robert Bell, and Chris Volinsky. "Matrix factorization techniques for recommender systems". In: *Computer* 42.8 (2009), pp. 30–37.

[18] Yiqi Gu et al. "Robust weighted SVD-type latent factor models for rating prediction". In: *Expert Systems with Applications* 141 (2020), p. 112885.

[19] Shijie Wang, Guiling Sun, and Yangyang Li. "SVD++ recommendation algorithm based on backtracking". In: *Information* 11.7 (2020), p. 369.

[20] Tongliang Liu and Dacheng Tao. "On the performance of manhattan nonnegative matrix factorization". In: *IEEE Transactions on Neural Networks and Learning Systems* 27.9 (2015), pp. 1851–1863.

[21] Steffen Rendle. "Factorization machines with libfm". In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 3.3 (2012), pp. 1–22.

[22] Shuai Zhang et al. "Deep learning for recommender systems". In: *Recommender systems handbook*. Springer, 2021, pp. 173–210.

[23] Huifeng Guo et al. "DeepFM: a factorization-machine based neural network for CTR prediction". In: *arXiv preprint arXiv:1703.04247* (2017).

[24] Aanchal Mongia et al. "Deep latent factor model for collaborative filtering". In: *Signal Processing* 169 (2020), p. 107366.

[25] Taleb Alashkar et al. "Examples-rules guided deep neural network for makeup recommendation". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 31. 2017.

[26] Gintare Karolina Dziugaite and Daniel M Roy. "Neural network matrix factorization". In: *arXiv preprint arXiv:1511.06443* (2015).

[27] Abebe Tamrat Tegene et al. "Deep Learning Based Matrix Factorization For Collaborative Filtering". In: *2021 18th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*. IEEE. 2021, pp. 165–170.

[28] Abebe Tegene et al. "Deep Learning and Embedding Based Latent Factor Model for Collaborative Recommender Systems". In: *Applied Sciences* 13.2 (2023), p. 726.

[29] Lei Zheng, Vahid Noroozi, and Philip S Yu. "Joint deep modeling of users and items using reviews for recommendation". In: *Proceedings of the tenth ACM international conference on web search and data mining*. 2017, pp. 425–434.

[30] Aminu Da'u, Naomie Salim, and Rabiu Idris. "An adaptive deep learning method for item recommendation system". In: *Knowledge-Based Systems* 213 (2021), p. 106681.

[31] George Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[32] Binbin Hu et al. "Leveraging meta-path based context for top-n recommendation with a neural co-attention model". In: *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 2018, pp. 1531–1540.

[33] Yehuda Koren. "Factorization meets the neighborhood: a multifaceted collaborative filtering model". In: *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2008, pp. 426–434.

[34] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).