Multimedia Tools and Applications (2024) 83:42125–42147
https://doi.org/10.1007/s11042-023-17029-7

# Integrating metadata into deep autoencoder for handling prediction task of collaborative recommender system

**Gopal Behara[1] · V. Ramanjaneyulu Yannam[2] · Anand Nayyar[3] · Dilip Kumar Bagal[4]**

Received: 1 March 2023 / Revised: 29 August 2023 / Accepted: 11 September 2023 /
Published online: 16 October 2023


**Abstract**
Nowadays, Deep learning (DL) techniques have been proven successful as learning techniques in various research fields ranging from computer vision to social networks. The approach of DL is flourishing in the field of recommender systems (RS). Researchers have deployed metadata or auxiliary information using DL approaches in diverse applications in the last decade to achieve better recommendation accuracy. Thus, the metadata plays a vital role in obtaining a better user-item interaction. At the same time, existing techniques are based on fixed user and item factors. Therefore, the model does not correctly identify actual latent factors representation, resulting in a high prediction error. To handle this problem, a user metadata embedding using a deep autoencoder RS model called "***Metadata Embedding Deep AutoEncoder (MEDAE)***" based collaborative filtering is proposed. MEDAE model takes embeds user metadata such as demographics along with the rating data. The MEDAE model consists of an embedding layer, Encoder, and Decoder. The embedding layer generates embedding or latent features of the users, items, and metadata; Encoder receives concatenated features of the user, item, and metadata, then encodes the inputs and passes them to the decoder; and the decoder reconstructs the output. To test the effectiveness of proposed model Root Mean Squared Error and Mean Absolute Error measures are used. Different architectures (like Big-Small-Big (BSB) (5), BSB (3), Small-Big-Small (3), and SBS (5)) of the MEDAE model are evaluated on MovieLens datasets along with different parameters such as activation functions (ELU and SELU) and regularization and results concluded that the MEDAE with SBS (3) and ELU + SELU component improves **4**% of RMSE and **2**% MAE over the baseline methods.

**Keywords** Collaborative Filtering · Recommender System · Deep Learning · Metadata · Autoencoder

## 1 Introduction

A recommender system (RS) is a sub-category of an information retrieval system that determines a particular item's ranking or user preference. RSs are a collection of methods that assist users with finding recommendation-based items. Recommendations or

Extended author information available on the last page of the article

suggestions are often geared toward assisting the decision-making process, whether choosing the next music to listen to, reading a news story, watching the next movie, or applying for a job. RS addresses these problems by filtering out the vast majority of irrelevant items in the catalog in order to display a few highly relevant items that the user may find interesting.

Nowadays, recommendation systems (RSs) play a vital role in e-commerce industry and online platforms such as social-networking, YouTube, Facebook, and etc. Big companies like Netflix, Flip-kart, Spotify, and Amazon use filtering techniques i.e., recommender systems (RSs) to recommend useful products to an intended user. It should be possible for the RS to predict user preferences based on previous interactions, such as viewing product pages called implicit feedback or leaving reviews for movies, called explicit feedback. Amazon, Flipkart's product recommendations, movie recommendations of Netflix, and Spotify's music suggestions are all examples of successful RS.

Therefore, the performance of the RSs has made huge impact on the success of these companies. According to the collection of data and the way these data being recommended, the approach of RSs can be classified into two parts: collaborative filtering (CF) and content based (CB) recommendations [1, 2]. CB techniques consider contextual data that is date, time, and location [3]. The disadvantage of the CB technique is that it usually requires detailed profiling of every item in the database, which is not always possible. Additionally, similarity computations are limited by domain knowledge as well as to the explicit characteristics (e.g., genres or directors) rather than more interactions between the attributes.

On the other hand, CF technique recommends items to the active users based previous history of users; hence CF approaches are well known for personalized recommendations. Specifically, CF technique analyze the user's interest and preferences of similar users and predict the items accordingly to the users. In designing RSs, the objective is to enhance the prediction accuracy of RSs. The prize contest of Netflix is an example of this problem [4], Netflix conducts a prize contest on the substantial improvement of prediction accuracy of the RS technique. This is a traditional collaborative filtering problem: Infer the unobserved entries in an interaction matrix of $R^{m \times n}$ whose $(i, j), j^{th}$ item given by $i^{th}$ user. The prediction performance is measured by RMSE.

Collaborative filtering is a widely used approach in recommender systems, which involves making recommendations to users based on the preferences of similar users. Deep collaborative filtering is a variant of this method that uses deep learning techniques to model user-item interactions.

However, there are several limitations to deep collaborative filtering. One limitation is that it requires large amounts of data to train the model effectively, which can be a challenge for smaller businesses or startups. Additionally, the model may suffer from overfitting if there are not enough diverse examples to learn from. Another limitation is that deep collaborative filtering is mainly based on the past behavior of users and does not account for changes in their preferences over time. This means that the model may not be able to make accurate recommendations if users' preferences change frequently. Furthermore, deep collaborative filtering may not be effective in recommending items that are outside of users' known preferences. For example, if a user has only interacted with a specific genre of movies, the model may not be able to recommend movies from other genres that the user might enjoy. Finally, while deep collaborative filtering has shown promising results in some applications, it is not without its limitations. To overcome these limitations, researchers are exploring new techniques such as incorporating contextual information and incorporating additional data sources to improve the accuracy of recommendations.

In recent years, Deep Learning (DL) has emerged as significant technique in image processing, computer vision, and various diverse fields of computer science [5]. The success of DL techniques motivates many researchers to deploy the DL approach in RSs to improve the prediction task of CF. The prediction task of RSs is measured by using several deep learning techniques. At the same time, auto-encoder (AE) based RSs are not considering the metadata of users or items. Thus, embedding user metadata into the deep AE RS model is of particular interest. Therefore, we first formulate the problem as follows and then design the model. The detail notations that are used in this article is shown in Table 1.

Let $U_{u=1}^{N}$ and $I_{i=1}^{M}$ be the N users and M items. Let O (u, i, $r_{ui}$) denote the log of the users' past interactions. The objective of RS is to suggest a list of items to the intended users according to his/her preferences. In several cases, O contains implicit data only that is the interaction $r_{ui}$ if exists; otherwise considered as missing. We denote $\widetilde{O}$ as a set of unobserved triples and be an augmented interaction pair that considers some data sampled from $O\prime$. Let $O_u$ represent the set of item choices of user $u$ in the training set, and $\widetilde{O}$ be the unknown choices of user u. Hence items in $\widetilde{O}_u$ are the candidates to be suggested to the user $u$. The goal of the RS is to suggest a subset of items from the candidate set to each user $u$. The estimated values are most likely to be 1. In some cases, the range of $r_{ui}$ between 1 to 5 or $\{0, 1\}$. In this paper, we consider the range of $r_{ui}$ 1 to 5.

Deep neural networks have been included into CF models with promising results to improve recommendation accuracy. Deep learning has transformed the area of recommendation systems in recent years.

### 1.1 Challenges in traditional collaborative filtering

It has proven effective to provide suggestions based on user-item interactions using traditional CF techniques like matrix factorization and neighborhood-based approaches. Making precise forecasts is difficult with these approaches due to data sparsity and the cold

**Table 1** List of notations

| Symbols or Notations | Descriptions |
|---|---|
| $U_{u=1}^{N}$ | Set of N users |
| $I_{i=1}^{M}$ | Set of M items |
| $O(u, i, r_{ui})$ | Log of user's past interactions |
| $\widetilde{O}$ | Set of unobserved tuples |
| $p_u$ | User latent factors |
| $q_i$ | Item latent factors |
| $\widehat{r}_{ui}$ | Estimated ratings of $u^{th}$ user to $i^{th}$ item |
| $J_\theta$ | Cost function or objective function |
| $S_{ij}$ | Similarity score of I and j item |
| $f(), g()$ *and* $h()$ | Activation functions |
| $W, \widetilde{W}$ | Weight vector at input an out layer |
| $b, \widetilde{b}$ | Biases |
| $X \in R^N$ | Aggregate features of user, item and user metadata |
| $L(x, x\prime)$ | Loss function |

start issue, which affects new goods or users with little to no interaction data. Furthermore, these techniques often provide poor suggestions since they find it difficult to recognize intricate patterns and semantic representations in the data.

### 1.2 Enhancements through deep autoencoders

Researchers have looked at integrating deep autoencoders (AEs) into the recommendation pipeline to solve the shortcomings of conventional CF approaches. AEs are strong unsupervised learning models that encode information into a latent space and then reconstruct it to develop an effective data representation. Collaborative filtering can catch complex user-item patterns and provide more precise suggestions by using the expressive potential of AEs.

### 1.3 Leveraging metadata for improved recommendations

The integration of information related to users and goods is another essential component of enhancing CF. To provide better suggestions, metadata such as user demographics, item features, and contextual data might offer useful insights. Particularly for users with little interaction data, adding information to CF models may help the system better manage the cold start issue and increase the accuracy of recommendations.

### 1.4 Research objectives

In this paper, we integrate deep autoencoders with metadata embedding to improve collaborative filtering for recommender systems. In order to capture rich feature representations, our suggested model, Metadata Embedding into Deep Autoencoder (MEDAE), learns from both user-item interaction data and auxiliary metadata. We want to solve the "cold start" issue and increase suggestion accuracy by adding information, even for people or things with few interaction data.

This work aims to increase the state-of-the-art in collaborative filtering-based recommender systems by integrating deep autoencoders with metadata, giving users more precise and tailored suggestions across a range of disciplines.

In this paper, the prediction task of collaborative filtering is investigated with the help deep structure of AutoEncoder. The main motivation behind this work to embed the metadata of the user such as gender, age and occupation along with the rating data in the proposed system. For this purpose, we design a Metadata Embedding Deep AutoEncoder (MEDAE) based CF model to enhance the prediction task of the CF based recommendations. Basically, MEDAE model receives an aggregate feature vector of user-items, and metadata, then estimates the predicted rating score to an item. In the encoder part, MEDAE uses a deep AutoEncoder (AE) to encode the interactions of user and items. Whereas the second part of the MEDAE model decodes as well as reconstructs the interactions or ratings given by the user to an item. The MEDAE is a generalization of several baseline models with a flexible structure. We deploy the idea of Deeprec [6]. In addition to this, we integrate user metadata by using deep structure of AE to improve the prediction task of CF.

### 1.5 Objectives of the paper

The following are the objectives of the paper:

1. To conduct background study and enlighten literature review with regard to Recommendation Systems (RS) and Autoencoders;
2. To propose novel methodology titled "Metadata Embedding Deep AutoEncoder (MEDAE)", to integrate user metadata into a Deep AutoEncoder, to improve the collaborative filtering prediction task;
3. To demonstrate and prove that MEDAE is a generalization of several baseline models but with a more flexible structure;
4. To test and validate the MEDAE model with different architecture such as BSB (512, 256, 512), BSB (512, 256,128, 256, 512), SBS (256, 512, 256) and SBS (128, 256, 512, 256, 128) along with average rating and default rating;
5. And, to compare the proposed model to know the impact of different components such as batch size, dropout and activation functions with various existing techniques.

### 1.6 Organization of paper

The rest of the article is structured as: Related work on applying neural network methods to RSs is enlightened in Sect. 2. Section 3 elaborates the materials and methods, that supports for designing the proposed method. Section 4 discuss about materials and methods and Sect. 5 describes our proposed model and learning algorithm in detail. Experimental results for the components analysis and performance comparisons are presented in Sect. 6. And finally, Sect. 7 concludes the paper with future scope.

## 2 Related works

In this section, we discuss similar literatures that are closely related to our proposed work.

Sedhian et al. [7] proposed a novel method called Autorec, where the authors used the autoencoder concept to encode the interactions and try to reconstruct partially observed ratings at the output layer. Wang et al. [8] used the Auto-Encoder for RS and improved the performance by replacing a Bayesian AE with a Topic Model component, which was utilized for learning the latent factor representations of the articles. A recurrent recommender framework was proposed by Wu et al. [9] and the Autorec model used a vanilla autoencoder (AE) structure and does not consider any metadata of users or items.

A loss function based on the corrupt input and reconstruction error was proposed by Strub and Jeremie in 2015 [10] to train stacked denoising auto-encoders (SDAE). Li et al. [11] decreased computational costs associated with deep learning by using a marginalized denoising autoencoder and discuss a hybrid deep CF (DCF) to tackle the sparsity of CF. The researchers studied the latent features and side information through the deep learning and improved the performance of CF. Wang et al. [12] proposed a hierarchical Bayesian technique called collaborative deep learning (CDL), which allows two-way interaction between content representation and collaborative filtering. Later, Collaborative deep ranking (CDR), an extended version of CDL, was proposed by Ying et al. [13]. Further, Wu et al. [14] improved the performance of the autoencoder-based

model by adding noise or corrupted inputs, such as collaborative denoising auto-encoder (CDAE). CDAE resolves top-n suggestions by integrating user bias and can generalize many CF techniques by examining pairwise and pointwise loss functions. Alameda-Pineda et al. [15] elaborated a non-linear matrix completion approach for the classification task by predicting unknown labels by considering rank minimization approach.

In a study by Zhang et al. [16], various types of transformations were proposed over multi-field categorical data, such as Factorization machines, RBM, and denoising AE. Kim et al. [17] applied CNN to detect localized features in text and images in recommendation systems and concluded that CNN overcomes the limitation of bags-of-words through weight filters. Zhang et al. [18] enhanced the RS model with the help of convolution, visual knowledge, and denoising auto-encoders. Zhang et al. [19] proposed an efficient hybrid collaborative filtering model via contractive auto-encoders titled "AutoSVD++" to capture implicit user feedback and meta information of item to enhance rating prediction of CF. Using deep neural networks (DNN), Volkovs et al. [20] learned a mapping from content to existing latent features to tackle the cold-start problem. In comparison, DNN produces a wide variety of mobile app suggestions, whereas linear model's overgeneralization generates irrelevant recommendations. Zheng et al. [21] portrayed user behavior using parallel convolutional neural networks on user-item feedback, before utilizing a final shared interaction.

He et al. [22] proposed neural collaborative filtering (NCF) to enhance the prediction task of CF and used multi-layer perceptron along with general matrix factorization (GMF) to find the interaction between the user and item and only considered user ID and item ID. The NFM approach was proposed by He & Chua [23] as a modification to the classical factorization machine by introducing a bi-interaction pooling layer. This modification enabled the extraction of low-dimensional feature interaction vectors, leading to impressive results. Nevertheless, the implementation of this technique requires side information during feature representation learning, which may be difficult to acquire in real-world scenarios.

He et al. [24] used NeuMF to incorporate Multilayer Perceptron (MLP) in place of dot product, which was commonly used in Matrix Factorization (MF) methods, to enable the learning of high-order matching functions. While MLP was capable of handling high capacity and nonlinearity, it was unable to capture low-order feature interactions. To overcome this limitation, NeuMF combined the strengths of both MF and MLP by merging their prediction vectors in the final layer. This allows for the learning of both linear and nonlinear matching functions. Shang et al. [25] introduced probabilistic MF with users' demographic information to address CF's sparsity; in their model, they observed different items with the same domain and consider the users with similar demographics across the domain. Further, Shang et al. [26] used a randomized latent factor model to handle the computational burden and improve CF's prediction accuracy. Yoon and Lee [27] used the word2Vec approach to enhance the performance of RS, but didn't use the autoencoder concept while learning the feature vector of user-item.

Wang et al. [28] developed an RS framework called neural graph collaborative filtering to exploit the interactions in graph structures during the embedding process but didn't used any metadata of either user or item. Xiao et al. [29] discussed the Bayesian deep collaborative MF to exploit the user-item latent factors from user's social interactions along with the rating matrix to alleviate the sparsity of CF. He et al. [30] introduced light GCN to learn user-item embeddings in the form of interactive graphs and also used the weighted sum of embedding as final embedding, but didn't used any auxiliary information. Pan et al. [31] developed a sparse stacked denoising AE to handle the data scarcity and imbalance

problem for social network. Further, authors included MF and deep learning framework to improve the accuracy of the RS.

Zhang et al. [32] deployed the attention based probabilistic MF technique to overcome the sparsity issue of the social RS. However, authors ignored the metadata and time-sensitive factor of rating data. Nisha and Mohan [33] included social information into a DL model to learn the social data with the help of embeddings. At the same time, authors neither considered the item's nor user's meta information, therefore, utilizing an auxiliary data in a deep learning model becomes challenging. Aljunid and Dh [34] discussed an efficient deep learning technique on collaborative RS (DCLRS) to handle the sparsity of CF. In their approach, authors transferred explicit rating data into implicit. At the same time, authors only considered ID of user-item at the training phase without incorporating any auxiliary data of users-items.

Nahta et al. [35] discussed metadata embedding using deep learning for handling the cold-start problem of CF and augmented user metadata and item genre with rating data and fed them to the deep structure of a neural network to estimate the prediction. Jena et al. [36] designed a neural model for movie recommender systems based collaborative filtering and used seven layers' neural architecture to estimate the rating prediction of an item for a particular user. Tahmasebi et al. [37] proposed a hybrid social movie RS by utilizing deep AE to alleviate the accuracy of movie RS and deployed each user's social influence to enhance the prediction of RS. Shambour [38] proposed a deep learning technique for multi-criteria RS and employed AEs to exploit the non-trivial and hidden representation among users w.r.t multi-criteria preference for more accurate suggestions. Yengikand et al. [39] discussed the Deep-MSR model to address the problem in dot products for describing the various latent factors. Authors exploited MLP and stacked AE to learn the item and user hidden features from the interaction matrix and concluded that including user preferences and item, features will outperform the baselines.

Behera and Nain [40] demonstrated a nonlinear non-negative MF model to tackle the sparsity issue of CF and extended this by integrating item metadata into the deep collaborative framework to improve the sparsity problem of CF [41]. Mercheri et al. [42] proposed a movie RS using auto-encoder to handle the data sparsity and then enhance the accuracy of RS using a pre-trained RBM model. Ratnakanth and Poonkuzhali [43] employed CF along with deep AE to predict users' ratings for suggesting certain tourist places in India to travelers. Liu and Wang [44] developed a novel RS for CF using dual AE called CFDA. The authors used dual AE to extract the latent factor of users and items and then minimized the error with the help of these representations.

Bougtub et al. [45] proposed a hybrid RS using a deep autoencoder to handle Cf's sparsity and scalability issues. In their approach, authors learned the user's preference through the auto-encoder and reconstruct the unknown ratings. Further, they applied SVD++to keep the correlation information between latent factors. Alagappan and Victor [46] presented an efficient graph-based web page RS to overcome CF's overspecialization and sparsity problem and deployed AE to cluster the preference vector of users based on visited web pages and suggested the web page with the highest score. Jalali and Hosseini [47] discussed a hybrid dynamic model for RS that utilized deep AE to accommodate the new user and item relationship over time. Mainly the authors deployed user ratings and social interaction between users to compute the similarity score for different users at different time stamps. Also, they thought that user behavior over time would help to update the similarity matrices, which decreases the error and increase user satisfaction. A movie recommendation with the temporal feature was proposed by Behera and Nain [48] to improve the personalization recommendation.

A unique technique for forecasting subsurface water level utilizing remote sensing photos was proposed by Stateczny et al. [49]. To obtain more accuracy than earlier techniques, an ensemble classifier and an enhanced hydro index value was deployed. A novel approach for enhancing the quality of medical photographs was presented by Ahmed et al. [50] by utilizing the MICQ unsupervised machine learning algorithm. A novel technique for cross-lingual handwritten character recognition employing an LSTM with elephant herding optimization was enlightened by Guptha et al. [51] and demonstrated that the proposed technique can outperform on a dataset of handwritten characters from several languages.

Using a hybrid features-based independent condensed nearest neighbor (ICNN) model, Praveena et al. [52] suggested a novel technique for content-based medical image retrieval (CBMIR). The authors demonstrated that their technique can reach state-of-the-art performance by evaluating it on a dataset of medical photos. Ahmed et al. [53] suggested a novel approach for enhancing medical picture quality utilizing an unsupervised machine learning algorithm known as MICQ. The authors demonstrated that their technique may greatly improve the quality of the photos by evaluating it on a collection of medical photographs. A novel approach for content-based image retrieval (CBIR) for histopathology pictures of liver cirrhosis was put forward by Guptha et al. [54]. To outperform earlier techniques, the authors combined an adaptive regularized kernel fuzzy C-means method with an earth mover's distance-based methodology.

An innovative technique for improving random forest was put out by Kamalalochana and Guptha [55] to identify illness in photos of apple leaves. To improve accuracy over earlier techniques, the authors used a genetic approach to adjust the random forest model's parameters. A novel strategy for identifying liver lesions using ROBUST machine learning was put out by Sundari et al. [56] The authors improved accuracy over existing techniques by combining features and machine learning algorithms. Using smartphones and a cheap hardware kit, Ahmed et al. [57] suggested a novel telemedicine strategy for remote patient monitoring. The authors demonstrate the efficacy of their strategy in raising the standard of care for patients in distant locations by evaluating it in a real-world context.

A thorough analysis of wireless technology for remote patient monitoring was presented by Nirmala et al. [58]. Authors illustrated the benefits and drawbacks of various technologies and provide suggestions for further study.

As compared to the literature elaborated above, the proposed model is a generic model that can embed user metadata using AE for the rating prediction task of a collaborative recommender system. At the same time, we adapt the deep structure of AE that can learn the metadata along with the user and item information for estimating rating of an item given by the user. Further we found that the baseline method such as deep-AE uses 0 as default rating, while auto-rec uses 3 as default rating. Therefore, we have tried different default rating from 0–5 and consider their average as final default ratings. Also, we observed that deep-AE uses largest layer in the middle and smaller at the beginning and the end that is SBS architecture. Whereas, Auto-rec uses big-small-big (BSB) architecture. Therefore, in our work we have used both architecture with different depth and found that the SBS (3) architecture with average rating as default rating and combination of SELU + ELU activation function performs better compared to other architecture and baseline methods.

## 3 Materials and methods

In this section, we explore the materials and background methods that support for designing the proposed methodology and conducting experiments.

**Table 2** Description of datasets

| Datasets | No. of items | No. of users | % of Sparsity |
|---|---|---|---|
| ML-100 K | 943 | 1682 | 93.69 |
| ML- 1 M | 3952 | 6040 | 95.81 |
| Amovies | 69,629 | 15,067 | 99.91 |

```
ratings = pd.read_csv(os.path.join(MOVIELENS_DIR, RATING_DATA_FILE),
                      sep='::',
                      engine='python',
                      encoding='latin-1',
                      names=['userid', 'movieid', 'rating', 'timestamp'])

ratings['user_emb_id'] = ratings['userid'] - 1
```

**Fig. 1** Code snippet of preprocessing rating data

## 3.1 Materials

### 3.1.1 Datasets

The datasets used for the experimental purpose are shown in Table 2. ML-100 K[1] dataset contains 100 K ratings in which 943 users provide ratings to 1682 movies within the range of 1–5. Each client has rated at least 20 items. 1 M[2] file has 1,000,209 (1 M) ratings of 3,900 films given by 6,040 users. Amazon Movies (Amovies)[3] dataset was bigger and sparser compared to the MovieLens datasets.

The sparsity indicates percentage of interaction in the database or in the rating matrix. Mathematically sparsity is defined in Eq. 1.

$$Sparsity = \frac{No.\ of\ users \times No.\ of\ items}{Total\ No.\ of\ ratings} \tag{1}$$

### 3.1.2 Data per-processing

The raw data file is separated by: without headers. We transform the raw data file into a CSV with headers, which can be easily imported using Pandas. All the user and movie ID will be subtracted by 1 for zero-based index. The snippet shows the preprocessing for rating data and similar preprocessing is applied to users' data and movies data (Figs. 1, 2).

The function data Preprocessor is used for this transformation. The init_value is the default rating for unobserved ratings. If average is set to True, the unobserved rating will be set as the average rating of the user.

[1] https://grouplens.org/datasets/movielens/100K/
[2] https://grouplens.org/datasets/movielens/1M/
[3] http://jmcauley.ucsd.edu/data/amazon/

| | user_emb_id | movie_emb_id | rating | timestamp |
|---|---|---|---|---|
| 0 | 0 | 1192 | 5 | 978300760 |
| 1 | 0 | 660 | 3 | 978302109 |
| 2 | 0 | 913 | 3 | 978301968 |
| 3 | 0 | 3407 | 4 | 978300275 |
| 4 | 0 | 2354 | 5 | 978824291 |

**Fig. 2** Snippet of rating data after preprocessing raw data

### 3.2 Methods

Most of the machine learning (ML) models have two parts: (a) model definition and (b) cost or objective function.

The first component formulates the relationship between the input and output. Whereas, the second component is to optimize the parameters and find the best parameters during training. Generally, RS models are defined as per Equation $\hat{r}_{ui} = f\left(W^T z + \tilde{b}\right) = \tilde{W}^T h\left(W^T \tilde{r}_u + X_u + b\right) + \tilde{b} = \tilde{W}^T\left(\sum \tilde{r}_{ui} W + X_u\right)$

$$\hat{r}_{ui} = f\left(W^T z + \tilde{b}\right) = \tilde{W}^T h\left(W^T \tilde{r}_u + X_u + b\right) + \tilde{b} = \tilde{W}^T\left(\sum \tilde{r}_{ui} W + X_u\right)$$
$$\hat{r}_{ui} = f\left(W^T z + \tilde{b}\right) = \tilde{W}^T h\left(W^T \tilde{r}_u + X_u + b\right) + \tilde{b} = \tilde{W}^T\left(\sum \tilde{r}_{ui} W + X_u\right)$$
$$\hat{r}_{ui} = f\left(W^T z + \tilde{b}\right) = \tilde{W}^T h\left(W^T \tilde{r}_u + X_u + b\right) + \tilde{b} = \tilde{W}^T\left(\sum \tilde{r}_{ui} W + X_u\right).$$

$$\hat{r}_{\{ui\}} = J_\theta(u, i) \tag{2}$$

where $\hat{r}_{ui}$ be the estimated rating of $u^{th}$ user on $i^{th}$ product, and $\theta$ represents model parameters that are to be learned during training. The function $J_\theta$ corresponds to the objective function that shows the behaviour of output on input. Some of the RS models reviewed in this section are- Latent Factor Model (LFM); Similarity Model (SM); Factorized Similarity Model (FSM) and SVD++.

#### 3.2.1 Latent Factor Model (LFM)

LFM model factorizes the interaction matrix $r_{ui}$ into user $\left(p_u\right)$ and item $\left(q_i\right)$ latent factors [1, 59, 60]

$$\hat{r}_{ui} = J_\theta^{LFM}(u, i) = p_u \cdot q_i^T \tag{3}$$

Further, hierarchical [61, 62] and factorize machine [63] models the $r_{ui}$ with help of side features of user/items.

### 3.3 Similarity Model (SM)

In SM [64], the interest of user for an item $i$ is modeled as the weighted sum of similarity score between $i^{th}$ and $j^{th}$ user and the preference for item $j$. Though it is similar to item-based collaborative filtering, however, it does not use item similarity techniques to find predicted scores, instead, SM learns a similarity score from data [65].

$$\widehat{r}_{ui} = J_{\theta}^{SM}(u,i) = \sum r_{ui} \cdot S_{ij} \tag{4}$$

#### 3.3.1 Factorized Similarity Model (FSM)

The main drawback of the SM models is impractical because of the quadratic relationship between parameters and items. To handle this a low-rank approximation technique that is FSM is used.

$$\widehat{r}_{ui} = J_{p,q}(u,i) = \left(\sum r_{ui} \cdot p_u\right)^T \cdot q_i \tag{5}$$

#### 3.3.2 *SVD++*

SVD++ [64] is the combination of FSM and LFM. This model considers implicit feedback as an additional feature along with explicit data such as ratings.

$$\widehat{r}_{ui} = J_{p,q}(ui) = \left(\sum r_{ui} \cdot p_j + p_u\right)^T \cdot q_i \tag{6}$$

## 4 Proposed Methodology: MEDAE

Inspired by deep autoencoder [6] we integrate the meta-features of the user into the deep autoencoder (MEDAE) and trained the model with much deeper. Figure 3 shows the architecture of the MEDAE model. To enable this, we use SELUs [66] activation function with a high dropout rate. Basically, an autoencoder (AE) is a neural network that has two units an encoder and a decoder. The encoder encodes the given input (x):$R^N \rightarrow R^K$ Whereas decoder(z) decodes z into $R^K \rightarrow R^N$.

The objective of AE is to achieve K latent space of data in such a way that the errors between input and output are minimized [67]. The AE with linear activation and code layer is capable to gain PCA [68] transformation in the encoder. However, our approach has both encoding and decoding parts with a feed-forward network such that it computes $l = g\left(W^T x + b\right)$ where $g$ is a nonlinear function.

In the forward pass, the MEDAE model takes augmented features vectors of users, items, and user metadata s.t $X = f(U,I,D)$ and $X \in R^N$, where $N$ denotes a number of movies. However, $X$ is very sparse matrix, while the output of the decoder, $f(X) \in R^N$ is dense and contains predictions for all movies in the corpus.

During training, an input $x \in R^n$ is very sparse because very few users have interacted with all product. Whereas, autoencoder's output $f(x)$ is dense. Let's consider an ideal scenario of $f$; such that $f(x)_i = x_i, \forall_i : x_i \neq 0$ and $f(x)_i$ accurately estimates all user's future preference for items $i : x_i = 0$. That is if a user interacts with a new item $k$ (i.e., generating a new vector $x\prime$) such that $f(x)_k = x'_k$ and $f(x) = f(x\prime)$. Therefore, the scenario, $y = f(x)$ should be a fixed point of a well-trained AE: $f(y) = y$.

To explicitly enforce fixed-point constraint and to be able to perform dense training updates, we augment every optimization iteration with an iterative dense re-feeding steps as follows:

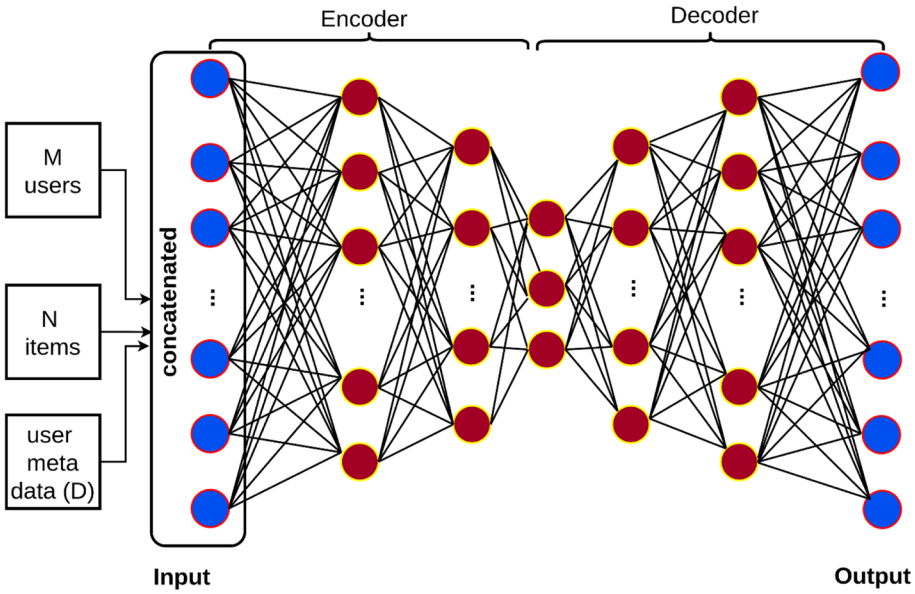


**Fig. 3** Illustration of Metadata Embedding Deep AutoEncoder

(1) In forward pass, compute dense $f(x)$ and loss according to Eq. 9. (2) In backward pass, compute gradients and perform weight update for each iteration.

### 4.1 Training MEDAE

During training the MEDAE model takes an input sample x (i.e., aggregate of user-id, movieid, and user metadata) and maps it into latent dimensions of $z$.

$$z = g\left(W^T x + b\right) \tag{7}$$

At the decoder end the model mapped back $z$ into the original input space to recapture the input vectors. Therefore, the output value is estimated as defined in Eq. 8.

$$\hat{r} = f\left(\widetilde{\mathrm{W}}^{\mathrm{T}} z + \widetilde{b}\right) \tag{8}$$

where the training consists of applying Adam optimizer to minimize the loss i.e., mean squared error as defined in Eq. 9.

$$L(x, x\prime) = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} x_i - x_i\prime^2 = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} x_i - \sigma\left(W z_i + b\right)^2 = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} x_i - \sigma\left(W\sigma\left(\widetilde{\mathrm{W}} x_i + \widetilde{\mathrm{b}}\right) + b\right)^2$$
$$\text{i.e.,} \underset{W, \widetilde{\mathrm{W}}, b, \widetilde{\mathrm{b}}}{argmin} \tfrac{1}{n} \sum L(x, x\prime) \tag{9}$$

**Input:** Rating matrix: $R^{M\times N}$, $\eta$, user matrix: $p_u \in P^M$, item matrix: $q_i \in Q^N$, user metadata information (age, job (j), gender (g))

**Output:** Predicted ratings: $\hat{r}_{ui}$

1 Random initialize uid, j, g, age, mid and genre embedding vectors.
2 Procedure MEDAE ($p_u$, $q_i$)
3 **Procedure** emb (uid, j, g, age, mid)
4 **for** *each user, j, g, age, mid* **do**
5 Find the embedding vectors of users:
6 Apply embedding layer to uid, j, g, age, mid and genre to generate embedding vectors
7 return embedding of uid, j, g, age, mid
8 end procedure
9 Concatenate the embedding vectors of uid, age, g, and j:

$$x = \phi(uid, age, g, j, mid) = \begin{bmatrix} uid \\ age \\ g \\ mid \end{bmatrix}$$

10 Train the AutoEncoder (x) that maps the sample x into z space
11 $z = g\,(W^T x + b)$
11 **Decoder(z)** maps back z into x
12 $\hat{r} = f\,(W^T z + \tilde{b})$
13 return $\hat{r}_{ui}$

**Algorithm 1:** Algorithm for MEDAE model

#### 4.1.1 Generalization of MEDAE with another model

MEDAE is a generalization of model-based CF models. The representation given in Sect. 3.2 can be treated as a special case of this architecture. Specifically, if we choose the identity or linear function for both $g(x)$ and $f(x)$ inputs. The output value of $\hat{r}_{ui}$ in Eq. 8 becomes:

$$\hat{r}_{ui} = f\left(W^T z + \tilde{b}\right) = \tilde{W}^T h\left(W^T \tilde{r}_u + X_u + b\right) + \tilde{b} = \tilde{W}^T \left(\sum \tilde{r}_{ui} W + X_u\right) \tag{10}$$

The detailed procedure for MEDAE is shown in Algorithm 1.

## 5 Experimentation, results and analysis

### 5.1 Experimental setup

We utilize standard datasets of Movielens in our experiments that are commonly used for validation of MEDAE. All datasets contain user metadata such as gender, age, occupation, and zip code. To test and validate the proposed approach, the system comprises of CPU with Intel Core 8$^{th}$ generation and NVIDIA RTX 3080 graphics. The experiment is done on UBUNTU 20.04 and uses Python 3.7 version with Keras. Further, we split the data into random 90%–10% train-test sets, and hold out 10% of the training set for validation. We used a fixed random_state = 999,613,182 for reproduction. When we

use train_test_split from sklearn, it is possible that reviews of one user are all split into one the training or test set and cause bias to avoid this we use Stratify with user_id. For example, if all the reviews of user 'A' are put into the training set, then during test time, there is no test data for this user. The test RMSE will be 0 for user 'A'. On the other hand, if all reviews are put into test set, then there is no review for this user during training time and cause the RMSE higher.

### 5.2 Performance metrics

We adopt two standard evaluation protocols that is the root mean squared error (RMSE) and mean absolute error (MAE) to measure the prediction error of the proposed approach. A low RMSE or MAE value indicates better results. The RMSE and MAE [69] are defined as follows:

$$RMSE = \sqrt{\frac{1}{T}\sum \left(r_{ui} - \hat{\mathrm{r}}_{\mathrm{ui}}\right)} \tag{11}$$

$$MAE = \sum \frac{\left(r_{ui} - \hat{r}_{ui}\right)}{T} \tag{12}$$

### 5.3 Techniques compared and results

Autoencoder [70] has been widely adopted into Collaborative Filtering (CF) for recommendation system. A traditional CF problem is inferring the missing rating in an $M \times N$ matrix R where $R_{ij}$ is the ratings given by the $i^{th}$ user to the $j^{th}$ item. The experimental outcomes are compared with baseline techniques discussed in Sect. 5.3.1.

#### 5.3.1 Techniques compared

The performance of the MEDAE model is compared against the following baseline models:

- RBM [71] Restricted Boltzmann Machine is a two-layer deep learning architecture that can be applied in RS.
- AutoRec [7] A novel AE model for collaborative filtering that has significantly improved the performance over RBM.
- Deeprec [6] is a deep AE based collaborative filtering; which introduce the deeper architecture for modelling the CF.
- CDAE [9] is a collaborative denoising AE for a Top-N recommendation
- SSAERec [72] It is a deep CF model that uses a stacked sparse AE into MF.
- NCF [22]: Neural collaborative filtering uses matrix factorization and multi layer Perceptron for finding the prediction task of recommender system.
- MEDCF [35]: Metadata embedding into deep collaborative filtering (CF) to improve the performance of CF based recommender system.

**Table 3** Performance comparison of MEDAE and benchmark models on MovieLens (ML) datasets

| Method | ML-100 K | | ML-1 M | |
|---|---|---|---|---|
| | RMSE | MAE | RMSE | MAE |
| Autorec [7] | 0.995 | 0.768 | 0.923 | 0.735 |
| RBM [71] | 0.965 | 0.738 | 0.874 | 0.686 |
| Deeprec [6] | 0.922 | 0.722 | 0.871 | 0.681 |
| CDAE [9] | 0.964 | 0.758 | 0.917 | 0.723 |
| SSAER [72] | 0.912 | 0.723 | 0.847 | 0.672 |
| NCF [22] | 0.931 | 0.728 | 0.868 | 0.669 |
| MEDCF [35] | 0.908 | 0.718 | 0.851 | 0.673 |
| DeepNMF [40] | 0.917 | 0.725 | 0.903 | 0.711 |
| MEDCRS [41] | 0.889 | 0.705 | 0.845 | 0.664 |
| MEDAE | **0.872** | **0.692** | **0.837** | **0.658** |

The proposed model (MEDAE) improves the prediction task due to the embedding of contextual information of the user by at least 2% and 2.2% compared to the state-of-the-art model in Tables 3 and 4, respectively

**Table 4** Performance comparison of MEDAE and benchmark models on Amovies dataset

| Methods | RMSE | MAE |
|---|---|---|
| Autorec [7] | 0.916 | 0.734 |
| RBM [71] | 0.892 | 0.691 |
| Deeprec [6] | 0.879 | 0.676 |
| CDAE [9] | 0.857 | 0.667 |
| SSAER [72] | 0.806 | 0.624 |
| NCF [22] | 0.808 | 0.629 |
| DeepNMF [40] | 0.815 | 0.632 |
| MEDCF [35] | 0.753 | 0.602 |
| MEDAE | **0.731** | **0.583** |

The proposed model (MEDAE) improves the prediction task due to the embedding of contextual information of the user by at least 2% and 2.2% compared to the state-of-the-art model in Tables 3 and 4, respectively

- DeepNMF [40]: it uses non-negative embedding features into the input layer of deep neural architecture to handle the prediction task of collaborative filtering.
- MEDCRS [41]: It embedded item metadata into the deep architecture of collaborative recommender system (CRS) to enhance the performance of CRS.

### 5.3.2 Results

Here, we discuss the quantitative analysis of the results in terms of Root Mean Square and Mean Absolute Error as discussed in Sect. 5.2 of the MEDAE model along with the state-of-the-art models.

Table 3 shows the quantitative result comparison of the MEDAE model and state-of-the-art models on MovieLens datasets.

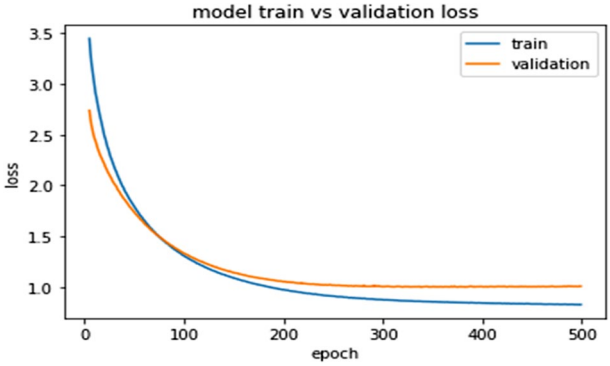


**Fig. 4** MEDAE model loss in terms of training and validation

It is found from Table 3 that the proposed model significantly improves the RMSE over the baseline models. That is MEDAE model improves the 4% and 2% of RMSE compared to the previous best model on ML-100 K and ML-1 M datasets respectively. We believe that MEDAE learns more features by embedding the metadata with the help of a deep autoencoder, hence the prediction task of the collaborative recommender system.

Similarly, Table 4 shows the performance comparison of MEDAE model with the baseline models. It is found that MEDAE model is improved RMSE of 2.2% and MAE of 2% over the best model i.e., MEDCF. Moreover, our model significantly outperforms compared to the baseline model due to embedding the metadata of users through the deep architecture of AE.

Further, we conducted the experiment with an average rating and zero as the default rating. It is observed that, when the model goes deeper, the zero default rating converged faster and with less noise. However, when we take a look at the loss graph depicted in Fig. 4 the gap between training and validation is larger in the zero default setting. This means when we use zero as the default rating, the model is easier to overfit.

### 5.3.3 Effect of batch size

Next, we verify the effectiveness of various batch sizes on the model performance as shown in Fig. 5. It can be noted that with increasing the batch size of the model improves the prediction accuracy significantly in terms of RMSE. Whereas Over-fitting may result from batch sizes that are too high, but too-small batch sizes may lead the network to converge slowly.

### 5.3.4 Effect of dropout

We examine the effect of different dropout ranging from 0.2 to 0.8 with a batch size of 256 and epochs of 500, experimenting the model with SBS architecture of n, 128, 256, 512, 256, 512, n. That is three layers in the encoder (128, 256, 512), a coding layer of 512 and three layers in decoder of 256, 512, n. However, we found that the model with

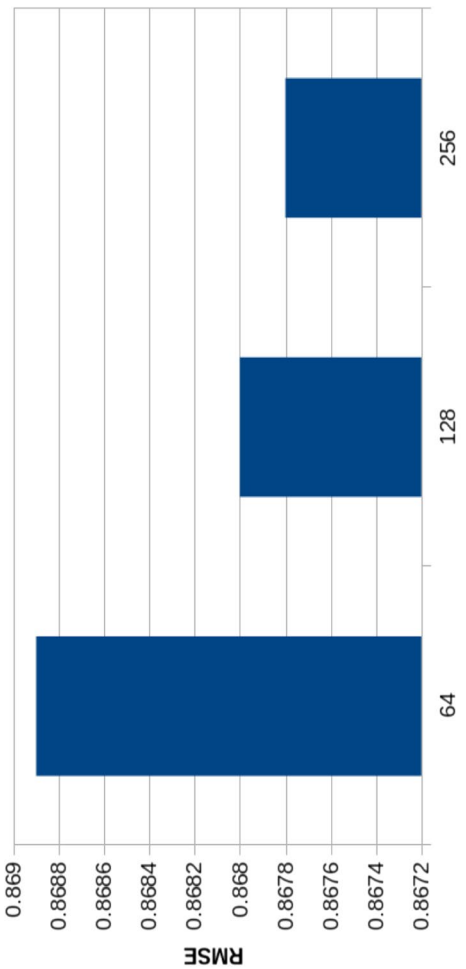


**Fig. 5** Effect batch size. A model with a 64 batch size clearly shows bad results. While models with a batch size of 128 and 256 result in RMSE of 0.8680 and 0.8607 respectively

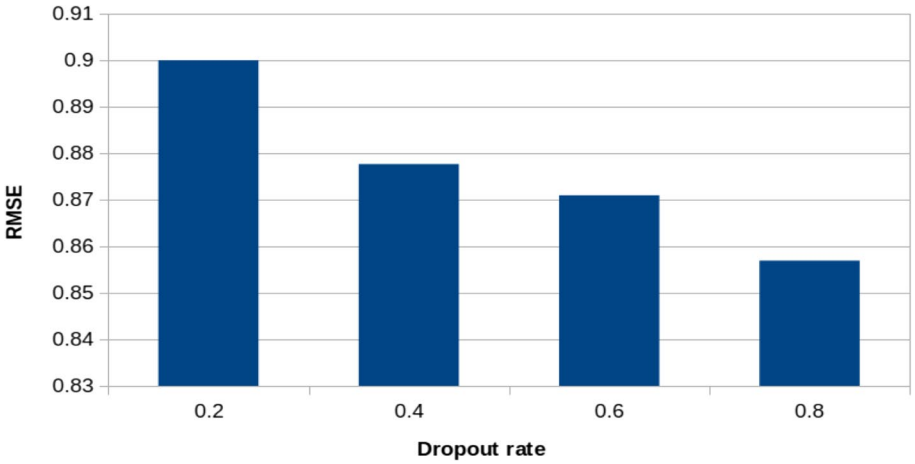


**Fig. 6** Effects of dropout. The model with a 0.2 dropout clearly shows a worse result. While MEDAE with drop probabilities of 0.6 and 0.8 result in RMSE of 0.8709 and 0.8569 respectively

**Table 5** Comparison of MEDAE with different architectures

| Architectures | Shape (depth) | No.of parameters | Test RMSE | Train RMSE |
|---|---|---|---|---|
| [512, 256, 512] | BSB (3) | 4.3 M | 0.856 | 0.746 |
| [512, 256, 128, 256, 512] | BSB (5) | 4.4 M | 0.869 | 0.827 |
| [256, 512, 256] | SBS (3) | 2.3 M | 0.857 | 0.760 |
| [128, 256, 512, 256, 128] | SBS (5) | 1.3 M | 0.868 | 0.840 |

this architecture, quickly over-fits if trained without regularization. To overcome this, we trained the model with several dropout values ranging from 0.2 to 0.8 and, interestingly, we found that the dropout rate of 0.8 gives the best among all. Figure 6 shows the performance of dropout probability w.r.t RMSE.

### 5.4 Result analysis/ Discussion

In this paper, we analyze the result in terms of following research questions (RQ):

**RQ1:** Does the MEDAE outperform state-of-the-art CF methods in terms of prediction accuracy?
The MAE and RMSE outlined in Table 2 reveals that the MEDAE model obtained the lowest MAE and RMSE value; hence our model outperformed the baselines on all datasets. This is because MEDAE incorporates metadata of users such as age, gender, and occupation to learn more features for obtaining the interaction of user-items. Whereas Autorec, RBM, Deeprec, and CDAE have not considered any auxiliary information. Compared to the AutoRec, RBM, and Deeprec baseline approaches,

**Table 6** Comparison of MEDAE with different architectures

| Default Rating | Activation Function | Test RMSE | Train RMSE |
|---|---|---|---|
| 0 | ELU, ELU | 0.856 | 0.746 |
| Average | ELU, ELU | 0.869 | 0.827 |
| Average | SELU, ELU | 0.857 | 0.760 |

MEDAE has a 12.26%, 9.23%, and 4.93% improvement of RMSE value respectively on ML-100 K. Whereas our model improves RMSE of 9.03%, 4.13%, and 3.83% compared to AutoRec, RBM on ML-1 M. Compared to the other baseline recommendation approach, MEDAE obtained better outcomes.

**RQ2:** Is the depth of the model architecture affects the performance of the model?

To justify the impact of the model architecture on the performance of the model, we conduct the experiment by considering different architectures as shown in Table 5 of the MEDAE model. We found that adding more layers will not help for both small-big-small (SBS) and big-small-big (BSB) shape rather the model gets over-fitted. Therefore, in our experiment, we consider SBS (3) because it produces fewer parameters to train a model and can mitigate over-fitting.

**RQ3:** How does the activation function affects MEDAE?

Similarly, to the impact of activation function on MEDAE, we conducted the experiment by considering different activation functions and the result as shown in Table 6 It is noticed that the MEDAE model obtains better results for a combination of SELU + ELU activation functions at the same time the default rating is set to average.

## 6 Conclusion and future work

In this paper, we propose a metadata embedding into deep Auto Encoder (MEDAE) architecture to enhance the prediction task of collaborative filtering. Specifically, we designed different architecture that is SBS and BSB with different depth such 3 and 5 of MEDAE. At the same time, we consider average rating as default rating along with ELU and SELU activation function. The MEDAE receives an aggregate feature of item, user and metadata of user and obtain the rating prediction task of a collaborative filtering-based recommender system. To know the effectiveness of the MEDAE model we conduct several experiments on Movielens datasets and observed that the MEDAE model with SBS (3) structure and SELU + ELU activation function outperforms the baselines at least **4%**.

Further, we will be extended the model by considering other auxiliary information such as knowledge base and temporal information. This information gains more attention in user's preference and makes better personalization. Similarly, we only considered the historical rating information along with the metadata of the user. Whereas we do not consider the short and long-term preferences of users. Therefore, this model can be extended by embedding the contextual state of long and short-term preference is another domain to enhance the personalization of the recommender system.

## Declarations

**Data availability** Authors declare that all the data being used in the design and production cum layout of the manuscript is declared in the manuscript.

**Conflicts of interest** The authors declare that they have no conflicts of interest to report regarding the present study.

## References


1. Behera G, Nain N (2022) Trade-off between memory and model-based collaborative filtering recommender system. In: Proceedings of the international conference on paradigms of communication, computing and data sciences: PCCDS 2021. Springer, Singapore, pp 137–146
2. Behera G, Nain N (2022) Gso-crs: grid search optimization for collaborative recommendation system. Sadhana 47(3):1–12
3. Adomavicius G, Tuzhilin A (2010) Context-aware recommender systems. In: Recommender Systems Handbook. Springer, Boston, MA, pp 217–253
4. Bennett J, Elkan C, Liu B, Smyth P, Tikk D (2007) Kdd cup and workshop 2007. ACM SIGKDD Explor Newsl 9(2):51–52
5. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
6. Kuchaiev O, Ginsburg B (2017) Training deep autoencoders for collaborative filtering. arXiv preprint arXiv:1708.01715
7. Sedhain S, Menon AK, Sanner S, Xie L (2015) Autorec: Autoencoders meet collaborative filtering. In: Proceedings of the 24th international conference on world wide web, pp 111–112
8. Wang H, Wang N, Yeung DY (2015) Collaborative deep learning for recommender systems. In: Proceedings of the 21st acm sigkdd international conference on knowledge discovery and data mining, pp 1235–1244
9. Wu CY, Ahmed A, Beutel A, Smola AJ, Jing H (2017) Recurrent recommender networks. In: Proceedings of the tenth ACM international conference on web search and data mining, pp 495–503
10. Strub F, Mary J, Philippe P (2015) Collaborative filtering with stacked denoising autoencoders and sparse inputs. In: NIPS workshop on machine learning for eCommerce
11. Li S, Kawale J, Fu Y (2015) Deep collaborative filtering via marginalized denoising auto-encoder. In: Proceedings of the 24th ACM International on conference on Information and Knowledge Management, pp 811–820
12. Wang H, Wang N, Yeung D-Y (2015)Collaborative deep learning for recommender systems. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pages 1235–1244
13. Ying H, Chen L, Xiong Y, Wu J (2016) Collaborative deep ranking: A hybrid pair-wise recommendation algorithm with implicit feedback. In: Pacific-Asia conference on knowledge discovery and data mining. Springer, Cham, pp 555–567
14. Wu Y, DuBois C, Zheng AX, Ester M (2016) Collaborative denoising auto-encoders for top-n recommender systems. In: Proceedings of the ninth ACM international conference on web search and data mining. ACM, pp 153–162
15. Alameda-Pineda X, Ricci E, Yan Y, Sebe N (2016) Recognizing emotions from abstract paintings using non-linear matrix completion. In: Proceedings of the IEEE conference on computer vision and pattern recognition. IEEE, pp 5240–5248
16. Zhang W, Du T, Wang J (2016) Deep learning over multi-field categorical data: –a case study on user response prediction. In: Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20–23, 2016. Proceedings 38. Springer International Publishing, pp 45–57
17. Kim D, Park C, Oh J, Lee S, Yu H (2016) Convolutional matrix factorization for document context-aware recommendation. In: Proceedings of the 10th ACM conference on recommender systems. ACM, pp 233–240
18. Zhang F, Yuan NJ, Lian D, Xie X, Ma,WY (2016) Collaborative knowledge base embedding for recommender systems. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. Chicago, pp 353–362

19. Zhang S, Yao L, Xu X (2017) Autosvd++ an efficient hybrid collaborative filtering model via contractive auto-encoders. In: Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval. ACM, pp 957–960
20. Volkovs M, Yu G, Poutanen T (2017) Dropoutnet: Addressing cold start in recommender systems. Advances in neural information processing systems, 30
21. Zheng L, Noroozi V, Yu PS (2017) 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, pages 425–434
22. He X, Liao L, Zhang H, Nie L, Hu X, Chua TS (2017) Neural collaborative filtering. In: Proceedings of the 26th international conference on world wide web, pp 173–182
23. He X, Chua TS (2017) Neural factorization machines for sparse predictive analytics. In: Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval. ACM, pp 355–364
24. He X, Liao L, Zhang H, Nie L, Hu X, Chua TS (2017) Neural collaborative filtering. In: Proceedings of the 26th international conference on world wide web (pp. 173–182).
25. Shang J, Sun M, Collins-Thompson K (2018) Demographic inference via knowledge transfer in cross-domain recommender systems. In: 2018 IEEE International Conference on Data Mining (ICDM), pp. 1218–1223. IEEE
26. Shang M, Luo X, Liu Z, Chen J, Yuan Y, Zhou M (2018) Randomized latent factor model for high-dimensional and sparse matrices from industrial applications. IEEE/CAA J Autom Sin 6(1):131–141
27. Yoon YC, Lee JW (2018) Movie recommendation using metadata based word2vec algorithm. In: 2018 International Conference on Platform Technology and Service (PlatCon). IEEE, pp 1–6
28. Wang X, He X, Wang M, Feng F, Chua TS (2019) Neural graph collaborative filtering. In: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, pp 165-174
29. Xiao T, Liang S, Shen W, Meng Z (2019) Bayesian deep collaborative matrix factorization. Proc AAAI Confer Artific Intell 33(1):5474–5481
30. He X, Deng K, Wang X, Li Y, Zhang Y, Wang M (2020) Lightgcn: Simplifying and powering graph convolution network for recommendation. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 639–648
31. Pan Y, He F, Yu H (2020) Learning social representations with deep autoencoder for recommender system. World Wide Web 23:2259–2279
32. Zhang W, Liu F, Xu D, Jiang L (2019) Recommendation system in social networks with topical attention and probabilistic matrix factorization. PLoS One 14(10)
33. C C N, Mohan A (2019) A social recommender system using deep architecture and network embedding. Appl Intell 49(5):1937–1953
34. Aljunid MF, Dh M (2020) An efficient deep learning approach for collaborative filtering recommender system. Procedia Comput Sci 171:829–836
35. Nahta R, Meena YK, Gopalani D, Chauhan GS (2021) Embedding metadata using deep collaborative filtering to address the cold start problem for the rating prediction task. Multimed Tools Applic 80(12):18553–18581
36. Jena KK, Bhoi SK, Mallick C, Jena SR, Kumar R, Long HV, Son NTK (2022) Neural model based collaborative filtering for movie recommendation system. Int J Inf Technol 14(4):2067–2077
37. Tahmasebi H, Ravanmehr R, Mohamadrezaei R (2021) Social movie recommender system based on deep autoencoder network using Twitter data. Neural Comput Appl 33:1607–1623
38. Shambour Q (2021) A deep learning based algorithm for multi-criteria recommender systems. Knowl-Based Syst 211:106545
39. Yengikand AK, Meghdadi M, Ahmadian S, Jalali SM J, Khosravi A, Nahavandi S (2021) Deep representation learning using multilayer perceptron and stacked autoencoder for recommendation systems. In: 2021 IEEE International conference on systems, man, and cybernetics (SMC). IEEE, pp 2485–2491
40. Behera G, Nain N (2022) DeepNNMF: deep nonlinear non-negative matrix factorization to address sparsity problem of collaborative recommendersystem. Int J Inform Technol 14(7):3637–3645
41. Behera G, Nain N (2022) Handling data sparsity via item metadata embedding into deep collaborative recommender system. J King Saud Univ-Comput Inform Sci 34(10):9953–9963
42. Mecheri K, Mamadji R, Klai S, Souici-Meslati L (2022) Enhanced deep autoencoder based recommender system. In: 2022 First International Conference on Big Data, IoT, Web Intelligence and Applications (BIWA). IEEE, pp 31–36

43. Ratnakanth G, Poonkuzhali S (2022) Indian tourist recommendation system using collaborative filtering and deep autoencoder. In: Information and communication technology for competitive strategies (ICTCS 2021) Intelligent Strategies for ICT. Springer Nature Singapore, Singapore, pp 341–356
44. Liu X, Wang Z (2022) CFDA: collaborative filtering with dual autoencoder for recommender system. In: 2022 International joint conference on neural networks (IJCNN). IEEE, pp 1–7
45. Bougteb Y, Ouhbi B, Frikh B, Zemmouri E (2022) A Deep Autoencoder-Based Hybrid Recommender System. Int J Mob Comput Multimed Commun (IJMCMC) 13(1):1–19
46. Alagappan JK, Victor SP (2023) Deep auto-encoder based clustering algorithm for graph-based web page recommendation system. Concurrency and Computation. Pract Exper 35(2):e7505
47. Jalali S, Hosseini M (2022) Collaborative filtering in dynamic networks based on deep auto-encoder. J Supercomput 78(5):7410–7427
48. Behera G, Nain N (2023) Collaborative filtering with temporal features for movie recommendation system. Procedia Comput Sci 218:1366–1373
49. Stateczny A, Narahari SC, Vurubindi P, Guptha NS, Srinivas K (2023) Underground Water Level Prediction in Remote Sensing Images Using Improved Hydro Index Value with Ensemble Classifier. Remote Sens 15(8):2015. https://doi.org/10.3390/rs15082015
50. Ahmed ST, Guptha NS, Lavanya NL, Basha SM, Fathima AS (2022) Improving medical image pixel quality using micq unsupervised machine learning technique. Malaysian J Comput Sci 53–64
51. Guptha NS, Balamurugan V, Megharaj G, Sattar KNA, Rose JD (2022) Cross lingual handwritten character recognition using long short term memory network with aid of elephant herding optimization algorithm. Pattern Recogn Lett 159:16–22. https://doi.org/10.1016/j.patrec.2022.04.038
52. Praveena HD, Guptha NS, Kazemzadeh A, Parameshachari BD, Hemalatha KL (2022) Effective CBMIR System Using Hybrid Features-Based Independent Condensed Nearest Neighbor Model. Hindawi J Healthcare Eng Article ID 3297316. https://doi.org/10.1155/2022/3297316
53. Ahmed ST, Sreedhar Kumar S, Guptha NS, AlShammari NK, Basha SM (2016) Improving Medical Image Pixel Quality Using MICQ Unsupervised Machine Learning Technique. J Comput Sci, University of Malaya 53–64. https://doi.org/10.22452/mjcs.sp2022no2.5
54. Guptha NS, Patil KK (n.d.) Detection of macro and micro nodule using online region based-active contour model in histopathological liver cirrhosis. Int J Intell Eng Syst 11(2):256–265
55. Kamalalochana N, Guptha NS (n.d.) Optimizing random forest to detect disease in apple leaf. Int J Eng Adv Technol 8(Issue 5):244–249
56. Sundari LS, Guptha NS, Shruthi G, Thanuja K, Anitha K (2019) Detection of liver lesion using ROBUST machine learning technique. Int J Eng Adv Technol 8(5):214–219
57. Ahmed SST, Thanuja K, Guptha NS, Narasimha S (2016) Telemedicine approach for remote patient monitoring system using smart phones with an economical hardware kit. In: 2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE'16). IEEE, pp 1–4
58. Nirmala SG, Thanuja K, Kiran Kumari PATIL (2014) Wireless Technology To Monitor Remote Patients-A Survey. International Journal of Computer Networking. Wireless and Mobile Communications (IJCNWMC) 4:65–76
59. Koren Y, Bell R, Volinsky C (2009) Matrix factorization techniques for recommender systems. Computer 42(8):30–37
60. Mnih A, Salakhutdinov RR (2007) Probabilistic matrix factorization. Advances in neural information processing systems 20
61. Agarwal D, Chen BC (2009) Regression-based latent factor models. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 19–28
62. Zhang L, Agarwal D, Chen B-C (2011) Generalizing matrix factorization through flexible regression priors. In: Proceedings of the Fifth ACM Conference on Recommender Systems, pp. 13–20
63. Rendle S (2012) Factorization machines with libfm. ACM Trans Intell Syst Technol (TIST) 3(3):1–22
64. 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, pp. 426–434
65. Ning X, Slim, GK Sparse linear methods for top-n recommender systems. In: Proceedings of the 2011 IEEE 11th International Conference on Data Mining, pp. 497–506
66. Klambauer G, Unterthiner T, Mayr A, Hochreiter S (2017) Self-normalizing neural networks. Advances in neural information processing systems 30
67. Hinton GE, Zemel R (1993) Autoencoders, minimum description length and helmholtz free energy. Advances in neural information processing systems 6
68. Hinton GE, Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. Science 313(5786):504–507

69. Yu J, Zheng X, Wang S (2019) A deep autoencoder feature learning method for process pattern recognition. J Process Control 79:1–15
70. Zhang G, Liu Y, Jin X (2020) A survey of autoencoder-based recommender systems. Front Comp Sci 14:430–450
71. Salakhutdinov R, Mnih A, Hinton G (2007) Restricted Boltzmann machines for collaborative filtering. In: Proceedings of the 24th International conference on machine learning, pp 791–798
72. Zhang Y, Zhao C, Chen M, Yuan M (2021) Integrating stacked sparse auto-encoder into matrix factorization for rating prediction. IEEE Access 9:17641–17648



## Authors and Affiliations

**Gopal Behara[1] · V. Ramanjaneyulu Yannam[2] · Anand Nayyar[3] · Dilip Kumar Bagal[4]**

✉ Anand Nayyar
anandnayyar@duytan.edu.vn

Gopal Behara
gbehera@gcekbpatna.ac.in

V. Ramanjaneyulu Yannam
ramanjaneyulu.yannam@gmail.com

Dilip Kumar Bagal
dilipbagal90@gmail.com

[1] Department of Computer Science & Engineering, Government College of Engineering Kalahandi, Bhawanipatna 766003, Odisha, India

[2] Department of Computer Science & Engineering, National Institute of Technology, Rourkela 569008, Odisha, India

[3] Graduate School, Faculty Information Technology, Duy Tan University, Da Nang 550000, Vietnam

[4] Department of Mechanical Engineering, Government College of Engineering Kalahandi, Bhawanipatna 766003, Odisha, India