Computer Modeling in Engineering & Sciences

Tech Science Press

Doi:10.32604/cmes.2025.063973

ARTICLE

A Deep Collaborative Neural Generative Embedding for Rating Prediction in Movie Recommendation Systems

Ravi Nahta[1], Nagaraj Naik[2,*], Srivinay[3] and Swetha Parvatha Reddy Chandrasekhara[4]

[1]Department of Computer Science and Engineering, Indian Institute of Information Technology Vadodara, Gandhinagar City, 382028, India
[2]Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, 576104, India
[3]Department of Information Science and Engineering, B.M.S. College of Engineering, Bengaluru City, 560019, India
[4]Department of Computer Science and Engineering, B.M.S. College of Engineering, Bengaluru City, 560019, India

*Corresponding Author: Nagaraj Naik. Email: nagaraj.naik@manipal.edu

Received: 31 January 2025; Accepted: 26 June 2025; Published: 31 July 2025

ABSTRACT: The exponential growth of over-the-top (OTT) entertainment has fueled a surge in content consumption across diverse formats, especially in regional Indian languages. With the Indian film industry producing over 1500 films annually in more than 20 languages, personalized recommendations are essential to highlight relevant content. To overcome the limitations of traditional recommender systems—such as static latent vectors, poor handling of cold-start scenarios, and the absence of uncertainty modeling—we propose a deep Collaborative Neural Generative Embedding (C-NGE) model. C-NGE dynamically learns user and item representations by integrating rating information and metadata features in a unified neural framework. It uses metadata as sampled noise and applies the reparameterization trick to capture latent patterns better and support predictions for new users or items without retraining. We evaluate C-NGE on the Indian Regional Movies (IRM) dataset, along with MovieLens 100 K and 1 M. Results show that our model consistently outperforms several existing methods, and its extensibility allows for incorporating additional signals like user reviews and multimodal data to enhance recommendation quality.

KEYWORDS: Cold start problem; recommender systems; metadata; deep learning; collaborative filtering; generative model

1 Introduction

Movies are an unavoidable component of both enjoyment and daily life. Because of their brief duration and widespread popularity, many people enjoy viewing them as a kind of entertainment. In the past, individuals had to physically visit movie theaters or wait for the movie to be televised in order to watch it. Streaming services like Netflix, Amazon Prime, and Hotstar have made it possible to watch movies on smart devices anytime and anywhere, and according to a survey conducted by MICA and Communication Crafts, the amount of time people spend on Video Streaming Apps in nations such as Australia, India, Indonesia, South Korea, and Thailand increased by 140% in 2018 compared to 2016. YouTube emerged as the dominant platform in India in 2018, with Amazon Prime and the Indian portals HotStar, JioCinema, and Voot following closely. In India, the number of people that view content via over-the-top (OTT) platforms is rather high [1].

Additionally, it has been documented that 97 percent of YouTube material is viewed in languages exclusive to specific regions, and 60 percent of YouTube viewing time takes place outside the six major metropolitan areas. Hoichoi, an all-Bengali content streaming platform, experienced a significant 85 percent

surge in traffic, with the number of unique visitors rising from 76,000 in March 2018 to 140,000 in March 2019. The survey conducted by consultancy firm KPMG India and OTT platform Eros Now reveals that Hindi is the most preferred language for content consumption among users in India, followed by Tamil, English, and other regional Indian languages. It demonstrates that the majority of Indian viewers choose to watch OTT content in regional languages. However, because of the unavailability of Indian Regional Movies (IRM) data, there has not been much research done on the IRM dataset for recommendations. The unique and first-of-its-kind IRM dataset used in this research was created by the IIIT Delhi institute.

One of our primary goals is to focus on the Indian movies dataset to be able to predict the ratings accurately by the users so that movies are recommended according to user personalization. There are several benchmarking datasets like Netflix, MovieLens, IMDB that are used for testing the recommendation system. The main drawback of these datasets is that they have very few Indian cinema movies listed as opposed to the Hollywood and English serials. They lack regional movies, and India has a diversified user list that spans about 20 regional movie languages and almost 1500–2000 movies released each year. India has a huge box office gross revenue of 2.1 billion US dollars each year as of 2015, which is third-largest in the world. Therefore, a dataset is must that is specific to Indian regional cinema to predict the ratings of the user for a better recommendation system. The number of users watches movies in India is most substantial than in other countries, which is easily verified that in 2011, 3.5 billion tickets were sold, which is 900 K more than Hollywood movies. The popularity of Indian movies is due to the film city Mumbai. The major film production cities include Mumbai, Chennai, Kolkata, Bangalore, Kochi, and Hyderabad. The Hindi movies are coined as Bollywood cinema, which has 43 percent box office revenue. The second-highest box office revenue comes from the South Indian Tamil and Telugu films. Another film culture is Bengali films, also known as Tollywood. India has a population of around 1.37 billion, as evidenced by the fact that it has around 2100 multiplex screens. Indian movies are watched worldwide and are a global enterprise that is spread in southern Asia and across North America, Asia, Europe, eastern Africa, the middle east, china, and elsewhere, with a reach in 90 countries. Therefore, as of now, we should have a recommender system specific for the Indian movies dataset that can predict ratings for diversified IRM.

Recommender systems utilize users' ratings for recommending items specific to the user's interests. Two approaches are used for recommender systems: collaborative filtering-based (CF) and content-based (CB). CB approaches exploit user and item profiles to recommend user-specific items. CF approaches rely on other users' preferences/ratings collaboratively and try to recommend items to users who have similar tastes to other users. CB approaches require users' implicit data, which is hard to capture against CF-based approaches, which mainly depend on other users' explicit data like ratings, which is easier to capture. One of the drawbacks of CF-based approaches is the well-known cold start problem [2,3], which can be resolved in CB approaches and using metadata information of users and items. Our work focuses on both CF and CB techniques to take advantage of both approaches.

Most popular websites like Netflix try to focus on Indian movie cinema, and the reason for that is to showcase the great stories of Indian movies worldwide. It is a great endeavor to enable people to know great regional and varied genres, and having different languages of Indian cinema. A robust approach is needed to capture the diversified Indian movies according to each user's preference. The recommendation accuracy can also be improved by correctly predicting the ratings provided by the user in the future. It can be realized that using old techniques like Matrix Factorization (MF) and other static methods, it can be hard to capture each user's taste. As deep learning techniques in various application domains are extensively used with excellent results, so using deep learning techniques can achieve the desired goal. We believe that applying a deep learning model to the IRM dataset and comparing it is the first of its kind. Also, as

of now, no recommendation model exists that is built on diversified Indian movies regional cinema for interested audiences.

The cold start problem is the primary issue with the recommender system, which is the inability to predict ratings correctly for a new user or a new item. The use of metadata as a side information plays a vital role in dealing with this problem effectively. Metadata information of the user or item gives additional information to the model whenever new user/item comes and therefore improving recommendation accuracy. While generating satisfactory recommendations using only a limited number of characteristics is possible, incorporating user and item metadata into the model can significantly enhance its effectiveness. With limited features, the model attempts to learn by generating arbitrary patterns. However, when extra supplementary features/information are provided, the acquired patterns become more logical. We conducted experiments to examine the impact of incorporating user and item metadata as auxiliary features in recommendation models on the accuracy of rating predictions.

Several deep generative models [4–6] have evolved which primarily differs in the way ratings were generated. They have shown impressive results and model the user-item latent vectors effectively to some extent. However, these generative model learns the latent vectors statically and compute the rating using a fixed inner-wise dot product between these latent vectors. Also, some generative models ignore the use of metadata features and the user-item abstraction process. The uncertainty in the ratings is also not captured intuitively. Also, they assume fixed user and item prior information, which limits the model's capability in learning the user-item interaction function. To deal with the above limitations, we propose a neural generative embedding (C-NGE) model that has the ability to generate both the latent vectors and the ratings by incorporating ratings and metadata features. For effective model learning we introduce noise in the form of metadata features, which helps in reducing overfitting of the model for the ratings given by the user towards the items. To make our model generative, we use a reparameterization trick wherein we move the process of sampling to an input layer so that the randomness in the function learnt is now associated with the noise and not inputs or the parameters of the model. It helps in learning and then generating the user-item embeddings dynamically. Deep neural networks are used to predict the final rating by learning the interaction function between the user and the item. The use of metadata features for users and items helps to alleviate the cold start problem.

The rest of this paper is laid out as follows. Section 2 describes the literature survey related to this work. Problem formulation and motivation are described in Section 3. Section 4 describes the proposed model in detail. Section 5 describes the experimental setup and results. Section 6 concludes the paper with possible future perspectives of this work.

2 Related Work

This section compiles a list of papers that use machine learning and deep learning techniques to solve recommendation problems. Here, papers are encouraged to include metadata and other auxiliary information, which is particularly useful for the recommendation task.

Recent advances in recommender systems have addressed information overload by leveraging deep learning techniques to enhance rating prediction and top-N ranking tasks [7]. Extant works further explore pre- and post-modeling challenges, benchmark datasets, feature learning strategies, and evaluation methodologies, collectively driving the development of next-generation recommendation models.

In the past, collaborative filtering (CF) techniques were extensively utilized for explicit user feedback, such as ratings and reviews. Currently, identical methods are being employed to examine implicit feedback, which includes a user's buying history, click-through rate (CTR), and webpage visit frequency. Collecting

and examining implicit input is a challenging endeavor. However, it is also highly valuable as it employs content-based methods that are beneficial for user tailoring and addressing the cold start problem. In order to mitigate the cold start problem, hybrid methods employ latent factor models in conjunction with content-based approaches.

Shahab et al. [8] presented a hybrid movie recommendation system that tackles the cold start problem by integrating Collaborative Filtering (CF), Singular Value Decomposition (SVD), and Generative Adversarial Networks (GAN). Content-based (CB) filtering enhances personalization by incorporating movie metadata like release year and genre. The approach, validated on the MovieLens dataset, demonstrates improved recommendation accuracy, especially for users with sparse interaction histories.

Siet et al. [9] proposed a deep learning-based movie recommendation system that addresses scalability, data sparsity, and the cold-start problem by leveraging user demographic information and sequential behavior. It employs a transformer with positional encoding and a Multilayer Perceptron (MLP) to model user preferences, while K-Means clustering enhances recommendation diversity. Evaluation of MovieLens datasets shows significant performance improvements over traditional methods in Top-N recommendations. Padmavathi et al. [10] proposed different movie recommendation models such as content-based, collaborative filtering (KNN, SVD, Boost), and hybrid models. Second, it shows the resilience of Matrix Factorization and benchmarks the results of the Netflix experiment. Numerical results indicate that the efficiency and accuracy of the hybrid models were better and, thus, that these methods are appropriate for real applications in practice.

Anwar et al. [11] presented Multi-Criteria Recommender Systems (MCRS) to enhance user satisfaction by incorporating multiple rating aspects into collaborative filtering. A modified similarity measure and user clustering technique are proposed to address sparsity and multidimensionality. Experimental results on benchmark datasets demonstrate improved accuracy and efficiency in rating prediction. Mao et al. [12] proposed a hybrid recommendation model that enhances matrix factorization by integrating weighted similarity measures and log-likelihood text mining to address sparsity and cold-start issues. It improves upon traditional Alternating Least Squares (ALS) by reducing information loss and error rates. Experimental results show superior performance with lower RMSE and higher F1 scores compared to existing models.

Sinha et al. [13] evaluated traditional machine learning models, Convolutional Neural Networks (CNNs), and Quantum Neural Networks (QNNs) for movie recommendations using the MovieLens-1M dataset. Two QNN architectures, leveraging quantum principles like superposition and entanglement, are introduced to improve accuracy. Results show that the simpler QNN model outperforms classical methods, reducing MAE and RMSE by 6%.

Auxiliary information in the form of metadata is used to improve the CF methods even more. Researchers [14] used metadata that considers item and user as vectors, which is an embedding of the original data in a multidimensional space. Meta-Prod2vec [15] is proposed for representing items uniquely by computing low-dimensional item embeddings using metadata and attributes to find item similarity interactions. For regularizing item embeddings, metadata is used as side information, and it leads to better recommendation tasks on the music dataset. For reducing the cold start problem, cross-domain recommendations are a practical approach that uses source domain rich information and applies it to the target domain, which has fewer user preferences. Previous work on cross-domain recommendations used the CF paradigm and neglected the content information of the items. Recently, a cross-domain recommendation [16] is proposed that uses the metadata of the movies like genres, directors, actors and themes, composers, and music styles metadata for the songs, and embed these metadata in collaborative matrix factorization to deal with the cold start problem. Using MovieLens and Netflix datasets to contain different metadata information [17] improves the quality of the recommendations. The Word2Vec embedding model [18] is

proposed to use metadata using deep learning for user personalisation in the movie rating prediction task. Performance improvement based on recall@100 is observed compared to the baseline approaches. Thereafter, research scholar coauthors and collaborators recommendation task as a link prediction task based on the knowledge graph embeddings [19] is proposed. They used scholarly metadata like citation networks and research publications. Different metadata views of the item from user reviews [20] are employed, which tells us about the item statistics, user opinions, and item quality. Attained results provide good rankings in contrast to their isolated versions. Many recommendation systems currently do not merely find patterns and then make recommendations based on user and item ratings. Many advancements have been made in the use of side information in conjunction with it to increase the quality of recommendation systems. To better comprehend the movie, a movie recommendation framework was created that included matrix factorization with extra visual features retrieved from pictorial data such as posters and still frames. Users may also wish to see a film based on appealing posters or frames, which cannot be predicted from reviews or ratings. As a result, features like colour histograms and predefined categories were employed to analyse user data and provide better recommendations [21].

Deep learning based recommender systems is an ongoing field of research that has vast applications in e-commerce and business scenarios. Table 1 summarizes that uses neural collaborative filtering techniques and metadata as auxiliary input information. The ACM RecSys 2018 conference series emphasized deep learning algorithms and techniques on recommender systems. Nowadays, generative models are gaining popularity for recommendation tasks to generate user-item associations using feedback. The authors [22] increase long-tail performances using adversarial training and Neural Collaborative Filtering (NCF). NCF learns the implicit feedback data associations, and simultaneously, the adversarial training is used to reproduce these associations to increase the long-tail performance. Users' feedback behavior many times contaminated due to imperfect preference selection [23]. This scenario can be handled by increasing the model's performance and robustness. They used adversarial training on collaborative autoencoder-based neural networks. The researchers exploited adversarial network embedding [24] to capture latent representations in learning robust and stable graph representations. RecGAN [25] model uses recurrent neural networks (RNN) and Generative adversarial networks (GAN) to model the short and long temporal behavior of the user and the item using GRU cells and learning latent features, respectively. The implicit hierarchical structure [26] of user and item preferences is exploited when they are not explicitly available. It is noticed that the hierarchy of user and item preferences improves the performance of recommender systems. Deep matrix factorization [27] models the implicit feedback information by embedding this feedback into a latent vector representation of user-item preferences to reduce the model parameters and therefore increase training efficiency. An attention-based latent factor model [28] is considered for an explainable personalized recommendation. The model gives attention weights to different item features based on that particular user's attention. For personalization, they modeled the attention distribution of every user based on his attention weights given to the item features. The existing problem of GAN-based CF is solved [29] by using vector-wise adversarial training that occurs in CF, which increases recommendation accuracy. Online social network information provides huge information beyond ratings [30]. So they proposed two methods. First, they used three sources of information, namely item reviews, ratings, and social relations, for rating prediction using latent factors and hidden topics. Secondly, using implicit feedback from ratings increases the capability and flexibility of the model. A model [31] is proposed that uses both short-term as well as long-term or session-based information for content-aware movie recommendation using adversarial learning. In this adversarial framework, the generator acts as a reinforcement learning agent that generates the next movie recommendation to the user sequentially, and the discriminator discriminates between the movies generated and the real movies recorded. They also incorporated posters of movies when ratings are sparse to increase

recommendation accuracy. The authors exploited multi-modal data [32] that has items with images and user reviews towards items to model preferences of the user and different aspects of item features with their importance. Then this aspect's importance is combined with the latent factor model that learns users and item latent factors. Reference [33] focused on aspect-aware ratings towards individual items. Finally, the actual rating linearly combines aspect ratings weighted by their aspect importance. It results in good interpretability of the model and alleviates the cold start problem. Embeddings were utilized to build two models, Neural Collaborative Filtering [34] and Neural Matrix Factorization, which directly represent user-item interaction matrices. In the context of movie recommendation, the suggested model was evaluated against advanced matrix factorization techniques like ItemPop and ItemKNN. The findings demonstrated a significant enhancement in performance. Autoencoders have also improved movie recommendations in a variety of ways. In comparison to existing methods, an adversarial training framework based on the Collaborative Denoising Autoencoder model improved both model robustness and overall performance. In the inputs, a small amount of perturbation is added to simulate corrupted or noisy data, which is subsequently reconstructed to the original noiseless input data as an output. Recommendation via dual autoencoders [35] was also developed, which used a gradient descent to learn hidden representations of users and movies using autoencoders. The model [36] proposed is a stacked denoising autoencoder-based hybrid model that performs deep learning of users and items' latent factors from side information, as well as collaborative filtering from the rating matrix. By combining stacked denoising autoencoder (SDAE) with matrix factorization, a hybrid recommendation model based on stacked denoising autoencoders used both a rating matrix and side information. These two models are combined to take use of their advantages in learning more expressive models [37]. More recently, the authors proposed the Meta Embedding Deep Collaborative Filtering (MEDCF) model [38] to address the cold start problem for the rating prediction task. However, the C-NGE model proposed in this work is different from the MEDCF model because the MEDCF model is not generative. Therefore, it cannot generate the abstract latent representations and is computed statically in contrast to the C-NGE model, where we used it in a probabilistic sense that can generate the latent representations dynamically.

Table 1: Summary of literature on movie recommender systems

Authors	Methods	Merit	Remarks
Shahab et al. [8]	Hybrid CF, SVD, GAN with CB filtering	Tackles cold-start with metadata-driven personalization	Improved accuracy for sparse user interaction history
Siet et al. [9]	Transformer + MLP + KMeans on user sequences	Enhances diversity and scalability	Outperforms traditional methods on Top-N metrics
Padmavathi et al. [10]	CF (KNN, SVD, Boost), CB and hybrid models	Hybrid and MF methods show robust accuracy	Validated on Netflix dataset with superior results
Anwar et al. [11]	MCRS with modified similarity + clustering	Reduces multidimensionality and sparsity	Improved prediction accuracy on benchmark datasets
Mao et al. [12]	ALS + weighted similarity + text mining	Reduces RMSE and cold-start effect	Hybrid model mitigates information loss in MF

(Continued)

Table 1 (continued)

Authors	Methods	Merit	Remarks
Sinha et al. [13]	QNNs vs. CNNs and classical machine learning (ML) models	QNNs outperform others in accuracy and efficiency	Achieved 6% lower MAE and RMSE on MovieLens-1M
Kula [14], Vasile [15]	Metadata embeddings, Meta-Prod2Vec	Improves item similarity and personalization	Enhanced vector representations for RS using metadata
Fernández-Tobias et al. [16]	Cross-domain CF using metadata	Leverages source-domain to resolve cold-start	Integrates metadata into matrix factorization
Yoon and Lee [18]	Word2Vec on metadata with deep learning	Improves recall in movie rating prediction	Achieved strong performance in recall@100
Henk et al. [19]	Knowledge graph embedding for link prediction	Applied to scholarly RS using metadata	Effective recommendation via academic citation networks
D'Addio et al. [20]	Multi-view metadata from reviews	Integrates quality, opinion, and stats	Outperforms isolated views in item ranking

3 Problem Formulation and Motivation

In this section, problem formulation is presented as a preliminary task, and the naive Matrix Factorization (MF) approach is briefly explained, with a highlight on the limitation of the inner dot product between the user and item latent vectors. Table 2 describes the notations list used in this work for clarity.

Table 2: Notation list. A user and an item are denoted by the subscripts u and i, respectively. The user metadata features are denoted by the subscript r, while the item metadata features are denoted by the subscript s. R-NGE and M-NE modules are denoted by the superscripts R and M, respectively

M	Number of users
N	Number of items
y_{ui}	Actual user u rating on item i
$\hat{y}_{ui}$	User u predicted rating on item i
y^R_{ui}	User u actual binarized score on item i for R-NGE
$\hat{y}^R_{ui}$	User u predicted score on item i for R-NGE
y^M_{rs}	Actual binarized score of user metadata r on item metadata s for M-NE
$\hat{y}^M_{rs}$	Predicted score of user metadata r on item metadata s for M-NE
I	Matrix of user-item interactions
I'	Binary interaction matrix of user-item metadata
z_u	User u latent vector
z_i	Item i latent vector

(Continued)

Table 2 (continued)	
z'_r	User metadata r latent vector
z'_s	Item metadata s latent vector
o_u^U	User u binarized one-hot encoded feature vector for R-NGE
o_i^I	Item i binarized one-hot encoded feature vector for R-NGE
o_r^R	User metadata r binarized one-hot encoded feature vector for M-NE
o_s^S	Item metadata s binarized one-hot encoded feature vector for M-NE
W	The neural network layer weight matrix
h	The output layer weights
b	Bias vector
a	Activation function

3.1 Matrix Factorization

Let u denotes the user and i denotes the item. We assume the total number of users and items as M and N, respectively. Let $I \in \mathbb{R}^{M \times N}$ be the implicit binarized interaction matrix which is constructed using the rating matrix as:

$$y_{ui} = \begin{cases} 1 & \text{if interaction (user } u, \text{ item } i) \text{ is positive;} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Here a value of $y_{ui} = 1$ indicates user u likes the item i. $y_{ui} = 0$ indicates that user u has not interacted with the item i.

Let the user latent vector be z_u and item latent vector be z_i. Using matrix factorization we can calculate the predicted rating $\hat{y}_{ui}$ as

$$\hat{y}_{ui} = f(u, i|z_u, z_i) = z_u^T z_i = \sum_{k=1}^{K} z_{uk} z_{ik}, \tag{2}$$

where K denotes the total dimensions of the latent space. This inner-wise dot product between latent vectors limits the model performance as it only finds the linear correlation between them as with any MF technique. A common solution to this limitation is to use large value of K but then it will not generalize the data properly due to overfitting and therefore, resulting in a large ranking loss. As a result, we will use the neural networks that learns the non-linear interaction function among the input data given.

4 The Proposed Model

This section describes a proposed model and algorithm using rating and metadata information for users and items through the deep generative framework and shows its comparison with the popular neural collaborative filtering (NCF) technique.

4.1 Collaborative Neural Generative Embedding (C-NGE)

The Collaborative Neural Generative Embedding (C-NGE) model is inherently more complex than traditional matrix factorization methods due to its multi-layered deep neural network architecture, which enables the capture of non-linear user-item interactions and metadata features. This complexity translates to significant computational requirements, necessitating powerful GPUs for efficient parallel processing during training, unlike standard CPUs commonly used in traditional methods.

The proposed Collaborative Neural Generative Embedding (C-NGE) model consists of three key modules: (i) Rating Neural Generative Embedding (R-NGE), which captures user-item interactions from the rating matrix, (ii) Metadata Neural Embedding (M-NE), which encodes user and item metadata, and (iii) Collaborative Neural Generative Embedding (C-NGE), which fuses both R-NGE and M-NE modules to enhance recommendation quality.

Each of these modules learns latent feature representations via deep neural networks (DNNs), ensuring higher-order feature extraction and abstraction.

In the R-NGE module, the user-item interaction matrix is represented as binary implicit feedback. We employ one-hot encoding followed by an embedding layer to extract latent features for both users and items. To prevent overfitting, we introduce sampling noise as metadata features, which helps generalize the learning process. However, direct backpropagation is challenging since the model samples latent vectors from a learned distribution rather than processing direct inputs. Specifically, the model estimates the mean (μ) and variance (Σ) from input features and then samples a latent vector (z) from a Gaussian distribution. Since this sampling step is non-differentiable, we utilize the reparameterization trick, where the noise component is explicitly sampled from a normal distribution and then transformed using learned parameters. This technique ensures the model remains differentiable, allowing effective gradient-based optimization.

The sampled latent vectors are then combined via an inner-wise dot product to produce a "pseudo-rating" ($\hat{y}_{ui}^{pseudo}$), which acts as an intermediate representation before final rating generation. This pseudo-rating helps in capturing meaningful interactions between users and items before integrating metadata features.

In the M-NE module, we encode user and item metadata as binary vectors and pass them through an embedding layer. The embedded metadata is processed using multiple layers of a deep neural network (DNN), with non-linear transformations (ReLU activations) applied to learn complex feature interactions. This module serves as a regularizer for the rating prediction process, improving generalization.

The C-NGE module is the final stage, where we concatenate the latent representations from R-NGE and M-NE to leverage both user-item interactions and metadata features simultaneously. This combined representation passes through deep neural interaction layers, capturing collaborative patterns in the data. The final output layer utilizes a sigmoid activation function to produce a probabilistic score, which represents the predicted user preference (rating) for an item. This probabilistic interpretation incorporates uncertainty, making the model more robust to noisy or sparse data.

By jointly learning from both rating and metadata information, the C-NGE model effectively generates user and item latent vectors that contribute to accurate and personalized rating predictions. The overall structure of our proposed model is illustrated in Fig. 1.

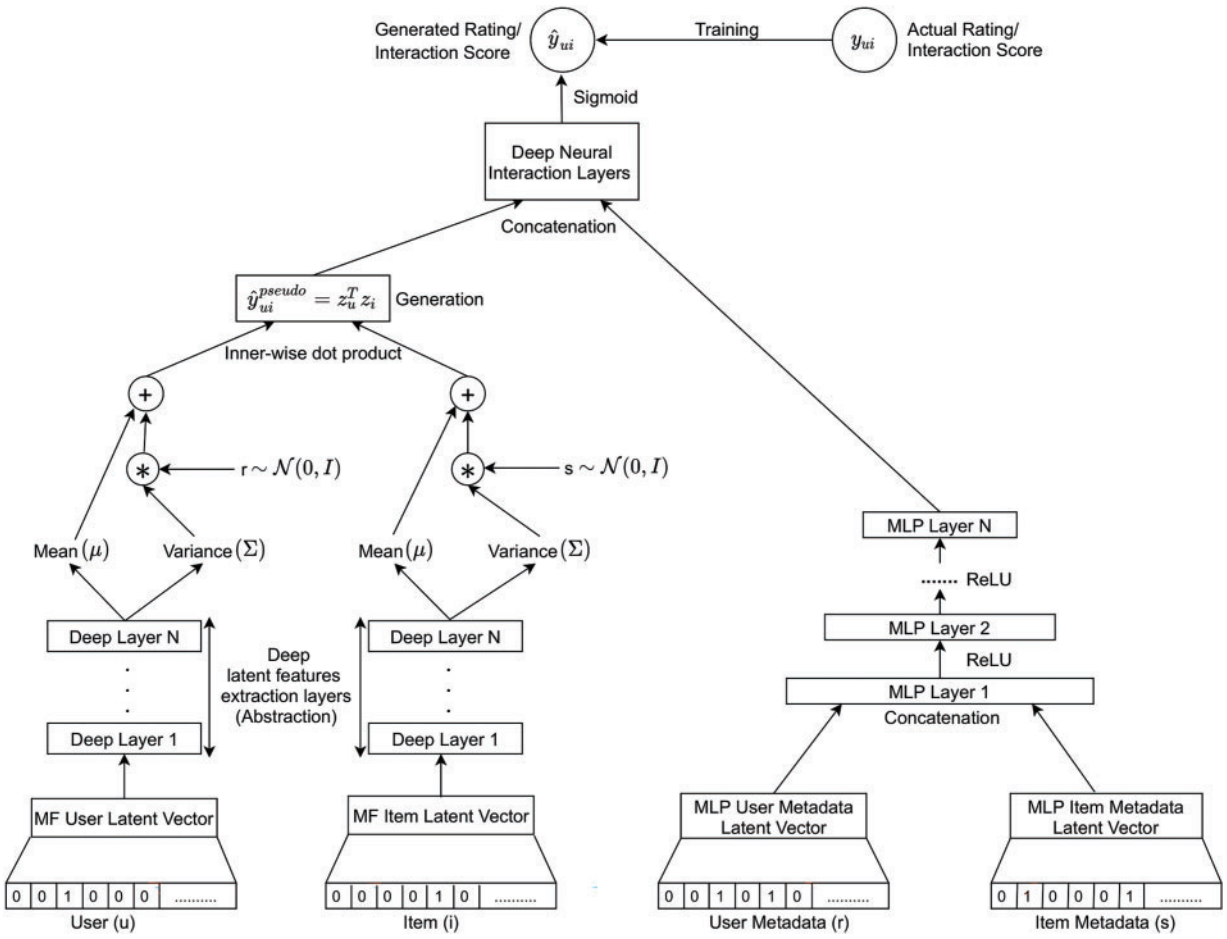


Figure 1: The proposed model

4.2 Difference between Proposed C-NGE Model and the NCF Technique

The proposed model is the extension and application of the popular NCF technique. The extensions are the following: (i) Incorporation of metadata features, (ii) the generative nature of the latent vectors as well as the rating using the reparameterization trick, and (iii) employing both abstraction and generation in a unified framework in a probabilistic manner to deal with the uncertainty in ratings. Therefore, the proposed model results in actual learning of the latent vectors and hence the generated rating as compared to the NCF technique. Incorporating metadata give neural networks additional information about user preferences towards items. Metadata is essential when the user has no rating for a specific item, and also when the new user has not rated any item, or a new item has no ratings. It avoids the cold start problem and, at the same time, avoids overfitting of the sparse rating/interaction matrix by acting as a regularizer. Metadata of users can contain user demographic information such as gender, age, and occupation. Metadata for the item (in our case, movies) may contain item-specific features such as genre, director, and cast. This information helps in understanding user preferences towards items and, therefore, increases the overall recommendation accuracy.

4.3 Algorithm Described Based on Proposed C-NGE Model

Let user metadata be r and item metadata be s. We define the $I' \in \mathbb{R}^{M \times N}$ interaction matrix between the user metadata and the item metadata features as an implicit binarized metadata features such that:

$$y_{rs} = \begin{cases} 1, & \text{if the interaction (user metadata } r \text{ and item metadata } s\text{) is observed;} \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The value $y_{rs} = 1$ denotes the interaction between r and s, otherwise 0. The implicit feedback has the limitation of indicating only the positive feedback between r and s while ignoring the negative feedbacks since 0 indicates either unobserved or missing entry.

Estimating the scores of missing entries in I' required to rank the items is defined as implicit feedback recommendation. Model-based approaches assume the use of the model itself to predict data. It can be formally defined as learning $\hat{y}_{rs} = f(r, s|\theta)$, where $\hat{y}_{rs}$ indicates the predicted score of y_{rs} and θ denotes the model parameters, and f indicates the function that maps the model parameters to $\hat{y}_{rs}$, identified as *interaction function*. Existing machine learning techniques that optimise a specific cost function are required to estimate model parameters θ. Pointwise loss and pairwise loss are the two most used cost functions. The M-NE module uses neural networks to calculate function f on the metadata binarized information in order to predict $\hat{y}_{rs}$. As a result, it inherently allows both pairwise and pointwise learning.

The model depicted in Fig. 1 is a layered neural network design. It has three modules as described earlier: R-NGE neural network is for interaction matrix, M-NE neural network is for metadata features and the C-NGE neural network is for collaborative embedding of the other two modules. The R-NGE neural networks generates the user and item latent vectors by learning through the rating interaction matrix using the reparameterization trick. To avoid overfitting of the model we use metadata features as the sampled noise so that the model learns better latent representations. The M-NE neural networks comprises of the deep neural layers representing the deep latent representations between user and item metadata features. The C-NGE neural network generates the final rating by learning through the interactions between the other two modules.

The R-NGE neural network calculates the predicted score $\hat{y}_{ui}^{R}$ by performing the inner dot product between the latent vectors of the user and item. The M-NE module employs a neural network structure to forecast the score $\hat{y}_{rs}^{M}$ (where M represents M-NE) by utilizing metadata information from users and things. In the C-NGE module, the values $\hat{y}_{ui}^{R}$ and $\hat{y}_{rs}^{M}$ are combined in the neural collaborative layers. To obtain the ultimate anticipated rating $\hat{y}_{ui}$, the combined output is inputted into a deep neural network, which further learns non-linear interactions between the user and object. The R-NGE module can serve as a versatile variant of the MF approach with the utilization of a linear activation function. The C-NGE neural network utilizes the linearity of R-NGE and the non-linearity of M-NE to describe user-item non-linear interactions. Denote o_r^R and o_s^S as binary one-hot encoded feature vectors representing the user metadata r and item metadata s, respectively. The content features are determined by the user's demographic data and the genre information of the item.

The M-NE module includes an embedding layer positioned above the input layer. A wholly connected layer transforms a sparse vector representation, encoded in a one-hot format, into a denser representation. The dimension of the last hidden layer determines the model's capacity, denoted as X. The computed value of the predicted score, denoted as $\hat{y}_{rs}^{M}$, is determined in the final output layer.

M-NE framework can be expressed mathematically as:

$$\hat{y}_{rs}^{M} = f(Z_r'^{T}, o_r^{R}, Z_s'^{T} o_s^{S} | Z_r', Z_s', \theta_f). \tag{4}$$

The matrices Z_r' and Z_s' represent the latent vectors for users and items metadata. Z_r' has dimensions $M \times K_1$ and Z_s' has dimensions $N \times K_2$, where M and N are the number of users and things, and K_1 and K_2 are the dimensions of the latent vectors. The latent dimensions for user and item metadata are correspondingly denoted by K_1 and K_2. The symbol θ_f represents the model parameter associated with the interaction function f. Let f denote a multi-layer neural network, which can be mathematically defined as:

$$f(Z_r'^{T}, o_r^{R}, Z_s'^{T}, o_s^{S}) = \phi_{out}(\phi_X(\ldots \phi_2(\phi_1(Z_r'^{T}, o_r^{R}, Z_s'^{T} o_s^{S}))\ldots)). \tag{5}$$

The mapping function for the output layer is ϕ_{out}, while the mapping function for the x-th M-NE layer is ϕ_x. There are a total of X M-NE layers.

Current pointwise methodologies for learning model parameters mostly focus on regression tasks, utilizing a mean squared loss function.

$$L_{sqr} = \sum_{(r,s)\in I' \cup I'^{-}} (y_{rs}^{M} - \hat{y}_{rs}^{M})^2. \tag{6}$$

In the binary metadata matrix for user-item interactions, I' represents the observed interactions, while I'^{-} represents the unobserved interactions with negative feedback.

Given this information, M-NE is designed as a probabilistic model that limits the model output $\hat{y}_{rs}^{M}$ to the range $[0, 1]$. This value represents the likelihood of user metadata r being relevant to item s. We considered a sigmoid activation function ϕ_{out} for the output layer. Let's utilize this configuration to depict the probability function in the following manner:

$$p(I', I'^{-} | Z_r', Z_s', \theta_f) = \prod_{(r,s)\in I'} \hat{y}_{rs}^{M} \prod_{(r,j)\in I'^{-}} (1 - \hat{y}_{rj}^{M}). \tag{7}$$

The negative logarithm of the above likelihood function is:

$$\begin{aligned} L &= -\sum_{(r,s)\in I'} log\hat{y}_{rs}^{M} - \sum_{(r,j)\in I'^{-}} log(1 - \hat{y}_{rj}^{M}) \\ &= -\sum_{(r,s)\in I' \cup I'^{-}} y_{rs}^{M} log\hat{y}_{rs}^{M} + (1 - y_{rs}^{M}) log(1 - \hat{y}_{rs}^{M}). \end{aligned} \tag{8}$$

The loss function is the target for minimization, and the Stochastic Gradient Descent (SGD) technique can be employed for optimization. The equation referenced as Eq. (8) is synonymous with the log loss or binary cross-entropy loss.

We compute the average and variability of the user and item embedded vectors, denoted as $\mu(z_u)$, $\sigma(z_u)$, $\mu(z_i)$, and $\sigma(z_i)$, to address the uncertainty in the ratings and provide more precise latent representations. Subsequently, we randomly select a z_u (or z_i) to transmit to the subsequent layer. Consequently, the entire process becomes unpredictable, and the function acquired by a neural network is no longer a continuous function of the inputs. In order to address this issue, we utilize a clever technique known as the reparameterization trick, wherein we transfer the sampling process to an input layer. To sample from a one-dimensional normal distribution with mean μ and standard deviation σ, we can generate a random variable ϵ from a standard normal distribution and then transform it using the equation $\epsilon \sim \mathcal{N}(\mu, \sigma)$. The

process involves generating a random variable from a standard normal distribution, denoted as $\mathcal{N}(0,1)$. This random variable is then used to compute an updated latent vector, $z_u = \mu + \sigma * \epsilon$, for the user (and a similar computation is done for the object). The function randomness is now linked to ϵ instead of being dependent on the inputs or parameters of the model. Our approach considers ϵ as two metadata characteristics, denoted as r and s. Following a normal distribution, these features are utilized as sampled noise in the R-NGE module. Subsequently, the mapping function of the initial neural R-NGE layer is delineated as follows:

$$\phi_1(z_u, z_i) = z_u \odot z_i, \tag{9}$$

where $\odot$ represents the element-wise inner product between the vectors z_u and z_i. The vector is then transformed to the output layer as follows:

$$\hat{y}_{ui}^R = a_{out}(h^T(z_u \odot z_i)), \tag{10}$$

where h and a_{out} are the weights and activation function of the output layer, respectively.

More accurately, the M-NE module under the C-NGE framework can be summarized mathematically as:

$$\begin{aligned} z_1 &= \phi_1(z_r', z_s') = \begin{bmatrix} z_r' \\ z_s' \end{bmatrix}, \\ z_2 &= \phi_2(z_1) = a_2(W_2^T z_1 + b_2), \\ &\cdots \\ z_L &= \phi_L(z_{L-1}) = a_L(W_L^T z_{L-1} + b_L), \\ \hat{y}_{rs}^M &= \sigma(h^T z_L). \end{aligned} \tag{11}$$

where b_x, a_x, W_x denote the bias vector, activation function, and weight matrix for the x-th layer of M-NE, respectively.

Coupling an R-NGE layer with a one-layer M-NE can be mathematically defined as:

$$\hat{y}_{ui} = ReLU(h^T a(z_u \odot z_i + W \begin{bmatrix} z_r' \\ z_s' \end{bmatrix} + b)). \tag{12}$$

The sharing of the embedding of R-NGE and M-NE may hinder the performance of the combined model by limiting their ability to utilize embeddings of the same length.

In order to enhance the versatility of the C-NGE model, distinct embeddings are utilized for each model, enabling them to learn their specific features more effectively.

In addition, the outputs of the last hidden layer for R-NGE and M-NE models are combined by concatenation, which is mathematically described as:

$$\begin{aligned} \phi^R &= z_u \odot z_i, \\ \phi^M &= a_L(W_L^T(a_{L-1}(\ldots a_2(W_2^T \begin{bmatrix} z_r' \\ z_s' \end{bmatrix} + b_2)\ldots)) + b_L), \\ \hat{y}_{ui} &= ReLU\left(h^T \begin{bmatrix} \phi^R \\ \phi^M \end{bmatrix}\right), \end{aligned} \tag{13}$$

where z_u represents the latent vector for user u, and z_r' represents the latent vector for user metadata r. Similarly, z_i represents the latent vector for item i, while z_s' represents the latent vector for item metadata s. The

activation function used for M-NE layers is the Rectified Linear Unit (ReLU). The integrated model is referred to as *C-NGE*, which simultaneously incorporates the non-linearity through a neural network structure. The system utilizes rating and metadata information to acquire knowledge of the latent representation of the user-item relationship. The learning algorithm for the rating prediction job in the C-NGE model is described in Algorithm 1. The reparameterization trick is employed in the C-NGE model to render the sampling process differentiable. Specifically, when the model samples latent vectors from a learned distribution (e.g., Gaussian), it encounters a non-differentiable step, which makes gradient-based optimization challenging. By reparameterizing the samples into a function of the distribution's parameters (mean and variance) and independent noise, we avoid this issue. The sigmoid activation function is chosen for the final output layer in the C-NGE model primarily because the output represents predicted ratings, which are typically treated as continuous rather than categorical variables. Sigmoid outputs a value between 0 and 1, making it well-suited for predicting probability scores or normalized ratings in a range suitable for interpretation as user preferences. In contrast, softmax is generally used for multi-class classification tasks where the outputs represent probabilities of distinct target classes summing to one. In the context of a rating prediction problem, where ratings can take continuous values or where we aim for a probabilistic interpretation, sigmoid is more appropriate, allowing each rating to be treated independently. In the R-NGE module, noise is incorporated to help generalize the model and to prevent overfitting by providing stochasticity during training. This specific noise formulation leverages metadata features as noise, which helps the model learn the latent representations of users and items more robustly by incorporating variability in the embeddings without introducing significant bias. This approach differs from other variational methods, which often rely on a specific distribution (e.g., Gaussian) and involve variational inference techniques to approximate a posterior distribution. While classical variational methods focus on estimating distributions to maximize evidence, the noise formulation used in C-NGE simplifies the representation learning process by directly infusing variability tied to metadata, thus enhancing the model's capacity to capture meaningful interactions without needing complete probabilistic modeling of latent spaces.

Algorithm 1: Algorithm for rating prediction using the proposed collaborative neural generative embedding (C-NGE) model

Input: $I \in \mathbb{R}^{M \times N}$, $I' \in \mathbb{R}^{M \times N}$, o_u^U, o_i^I, o_r^R, o_s^S

Output: Predicted rating $\hat{y}_{ui}$

procedure BinarizedRatingInteractionVector (o_u^U, o_i^I)

  for every o_u^U and o_i^I **do**

    Using embedding layer on o_u^U and o_i^I to produce z_u and z_i, respectively

    Using R-NGE layer to calculate $\hat{y}_{ui}^R \leftarrow z_u \odot z_i \leftarrow z_u^T z_i$

  end for

end procedure

procedure binarizedmetadatafeaturesvector (o_r^R, o_s^S)

  for every o_r^R and o_s^S **do**

    Using embedding layer on o_r^R and o_s^S to produce z_r' and z_s', respectively

    Using final M-NE layer to calculate $\hat{y}_{rs}^M \leftarrow \sigma(h^T a_L(W_L^T(a_{L-1}(\ldots a_2(W_2^T \begin{bmatrix} z_r' \\ z_s' \end{bmatrix} + b_2)\ldots)) + b_L))$

  end for

end procedure

procedure NeuralCollaborativeEmbedding (z_u, z_i, z_r', z_s')

(Continued)

Algorithm 1 (continued)

$\phi^R \leftarrow z_u \odot z_i$

$\phi^M \leftarrow a_L(W_L^T(a_{L-1}(\ldots a_2(W_2^T \begin{bmatrix} z'_r \\ z'_s \end{bmatrix} + b_2)\ldots)) + b_L)$

Using C-NGE layer to merge R-NGE and M-NE outputs and calculate for final rating prediction as:

$\hat{y}_{ui} \leftarrow ReLU(h^T \begin{bmatrix} \phi^R \\ \phi^M \end{bmatrix})$

end procedure

return $\hat{y}_{ui}$

5 Experiments and Results

In this section, the experimental setup is described, and an in-depth analysis based on the obtained experimental results is provided.

5.1 Datasets

Two datasets, MovieLens and Indian Regional Movies (IRM), have been used in our work, which are described below:

(1) MovieLens. MovieLens 100K[1] (ML100K) and MovieLens 1M[2] (ML1M) datasets were developed by the grouplens research project at the University of Minnesota. These datasets contain explicit ratings from users on items ranging from 1 to 5.

(2) IRM. Movies from all regions across the world can be found in popular datasets like IMDB, MovieLens, and FilmTrust. They do not, however, contain a lot of information about Indian regional films. Therefore, the IRM dataset [39] is also used in this work. It is the first Indian regional cinema dataset. The IRM dataset can be accessed from https://goo.gl/EmTPv6 (accessed on 25 June 2025). The IRM dataset contains 2851 movies, 919 users with 10,000 ratings in 18 different Indian regional languages. To make the user's process of rating multiple movies easier, the ratings have two categories, 1 or 0, which corresponds to liked or not rated. User metadata was collected while signing up through a web portal. Movies metadata has been scraped from IMDB, which consists of one of the most extensive collections of movies data. The following metadata information is associated with every movie (readers are encouraged to refer [39] for the detailed description on the IRM dataset):

(a) Movie id: Every movie is associated with a unique ID.
(b) Description: Synopsis of the movie.
(c) Languages: It is possible that a film was released in multiple regional languages.
(d) Release date: The movie's release date.
(e) Rating: To judge the movie's success, according to IMDB.
(f) Crew: Movie director, writer, and cast.
(g) Genre: A movie may have one or more genres out of the 20 different genres available on IMDB.

It is essential to include user metadata (demographic information) which influences user ratings for better recommendations. User has the following demographic information:

(a) User id: Every user is associated with a unique ID.
(b) Languages: The languages that the user knows.

[1] https://grouplens.org/datasets/movielens/100K/ (accessed on 25 June 2025).
[2] https://grouplens.org/datasets/movielens/1M/ (accessed on 25 June 2025).

5.1.1 Datasets Pre-Processing

The dataset contains a lot of information, some of which may not be usable for rating generation. Therefore, before the dataset can be used in the proposed model, it must be pre-processed. User metadata and item metadata are used as features in the proposed model. From user metadata, user id, gender, languages, and occupation are selected. For movie metadata, movie ID, languages, and genre are selected.

The dataset included both languages and genres in the form of a list. Therefore, each language and genre was segregated first. Now, for the movies dataset:

(i) Each movie was associated with all of the languages in a separate dataset created. Each language was encoded as either 1 or 0, depending on whether or not the movie was available in that language.
(ii) Another dataset was constructed in which each movie was associated with all of the genres. Each genre was encoded as either 1 or 0, depending on whether the movie belongs to that genre or not.
(iii) The movie ID was then used to combine the above two datasets.

For the users' dataset:

(i) Each user was associated with all of the languages in a separate dataset created. If the user prefers to watch a movie in that language or not, each language was encoded as either 1 or 0, respectively.
(ii) Gender and occupation were one-hot encoded using a label encoder and added to a separate dataset.
(iii) The user id was used to combine the above two datasets.

The two joined datasets (one for users and the other for movies) were joined once more to form a single dataset. The proposed model was implemented using this final pre-processed dataset.

5.2 MovieLens versus IRM Dataset

IRM dataset for recommender systems is developed at IIIT Delhi institute and is the first of its kind. It serves as a benchmark dataset for Indian regional cinema that has diversified languages, regions, and genres. With the popularity of Indian cinema worldwide and a massive number of movies released per year, it is essential to have this dataset so that further experiments can be performed.

MovieLens is a popular website that recommends movies to users. The MovieLens dataset inspired the IRM dataset. The Department of Computer Science and Engineering at the University of Minnesota, under the Grouplens project, developed the MovieLens dataset. The dataset has the user preferences in the form of ratings. This dataset was publicly released in several versions, such as 100 K ratings in the year 1998, 1 M ratings in the year 2003, and 10 M ratings in the year 2009.

Table 3 shows a comparison of the IRM, MovieLens 100 K, and MovieLens 1 M datasets.

Table 3: Comparison of datasets

Dataset	Users	Items	Ratings	Sparsity	Year
MovieLens 100 K	943	1682	100000	93.70%	1998
MovieLens 1 M	6040	3952	1000209	95.80%	2003
Indian Regional Movies	919	2851	10000	99.96%	2017

5.3 Baseline Methods

The proposed C-NGE model is evaluated and compared to the following baseline methods:

- **Item-Item Similarity** For model building, item-item similarity detects similarities between all item pairings. Multiple possibilities exist for item pairs to be similar. One such method is cosine similarity. Similar arguments apply to user-user CF. It is less popular than item-item collaborative filtering.
- **Matrix Factorization (MF)** Reference [40] approach generates latent features by multiplying two types of entities. MF is used in collaborative filtering to determine user-item relationships. We want to estimate how consumers would evaluate shop items based on user ratings so they can get recommendations.
- **Blind Compressed Sensing** References [41,42] calculate user and item latent factor matrices. The user latent factor matrix may be dense compared to the item latent factor matrix because the user-item interaction matrix is sparse. Sparsity in the item latent factor matrix improves recommendation accuracy.
- **Matrix Completion** In Reference [43], by filling in the unknown elements in a rating matrix with users as rows, objects as columns, and entries as ratings, the collaborative filtering problem can be addressed as a matrix completion problem. A prominent method for solving the above problem is the nuclear-norm-regularized (NNR) matrix.
- **Singular Value Decomposition (SVD)** Principal Component Analysis (PCA) is a prevalent linear algebra methodology extensively used in machine learning to reduce the dimensionality of data [44]. The Singular Value Decomposition (SVD) is a method for factorizing a matrix that effectively decreases the dimensionality of a dataset from N to K (where K is less than N).
- **Probabilistic Matrix Factorization (PMF)** Rating prediction is a challenge involving collaborative filtering (CF) to forecast ratings [45]. It can handle enormous datasets because its processing capacity increases proportionally with the number of examples in the dataset. It can be considered a probabilistic version of the SVD model.
- **Convolutional MF (ConvMF)** [46] Convolutional transform to filter non-pertinent information and integrate their learned latent profiles with probabilistic matrix factorization via rating-count distribution to reduce noise in the shared latent space.
- **Neural Collaborative Filtering (NCF)** [47] is a unified framework for implementing the MF method with neural networks using implicit feedback to learn the interaction function using the NeuMF neural network by concatenating Generalized Matrix Factorization (GMF) and Multilayer Perceptron (MLP) networks.
- **Supervised Matrix Factorization** [48]. Predicting ratings is challenging because of the sparse nature of the dataset. Utilizing data on user demographics and item categories can enhance the precision of predictions [49–52]. Users are categorized by age, gender, and occupation. Class label information is essential for learning user and movie latent feature vectors in supervised learning. Class label knowledge restricts the search space, minimizing problem identification.

5.4 Evaluation Metrics

We performed k-fold cross-validation to assess the model's performance across different subsets of the dataset. This allows us to validate the model on unseen data during the training process, providing a more reliable estimate of its generalization capability. In order to assess the efficacy of our model compared to previous baseline techniques, we employed two widely accepted metrics: Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). The Root Mean Square Error (RMSE), as described in [53], is a widely used metric for evaluating the accuracy of projected ratings. It is a standard quantitative measure commonly employed in supervised regression applications. The RMSE quantifies the disparity between the projected and actual rating. The root mean square error (RMSE) loss is computed using mathematical calculations:

$$RMSE = \sqrt{\frac{\Sigma(\hat{r}_{ui} - r_{ui})^2}{\#\text{ of ratings}}} \tag{14}$$

In error terms, MAE [54–56] does not impart any bias to extrema. Outliers or big error terms will be weighed equally with the other predictions if they exist. As a result, MAE should be preferred for evaluating rating accuracy when the importance of outliers is not a concern. The absolute difference between the actual and predicted scores is calculated. Mathematically, it is computed as:

$$MAE = \frac{\Sigma|r_{ui} - \hat{r}_{ui}|}{\#\text{ of ratings}} \tag{15}$$

For better recommendation accuracy, both RMSE and MAE are losses that should be minimised.

5.5 C-NGE Variants

We have used two variants of the proposed C-NGE model as follows:

(1) C-NGE (fixed latent vectors)

In this C-NGE variant, we remove the generative process using the reparameterization trick so that the latent vectors learned are fixed and therefore, ratings are predicted and not generated.

(2) C-NGE (without metadata features)

In this C-NGE variant, we ignore user and item metadata features and therefore deal only with the rating interaction matrix.

(3) C-NGE (with metadata features)

This is our proposed C-NGE model, which is used throughout the experiments as it allows us to embed metadata features using neural networks and also deals with the generation of the latent vectors and the ratings.

As indicated by the errors in Table 4 and Fig. 2, the C-NGE model showed remarkable improvements in comparison to the baseline methods. C-NGE with metadata features showed the least error (MAE (0.3088) and RMSE (0.5509)). It is because if no metadata is provided, the model will arbitrarily find the latent factors on its own, which may or may not result in useful recommendations. When metadata is included as a side feature in the model, however, the values are fixed but accurate, resulting in a more efficient model. It shows that adding metadata in a recommendation model does provide leverage over models with no metadata information.

Table 4: Loss evaluation of recommendation models on IRM dataset

S. No.	Techniques	MAE	RMSE
	Baseline methods		
1.	User-User similarity	0.5307	1.0312
2.	Item-Item similarity	0.6482	1.049
3.	Matrix factorization	0.471	0.9713
4.	Probabilistic matrix factorization	0.4811	0.9372
5.	Blind compressed sensing	0.4632	0.9612
6.	Matrix completion	0.4827	0.9264

(Continued)

Table 4 (continued)

S. No.	Techniques	MAE	RMSE
	Baseline methods		
	Our proposed model variants		
7.	C-NGE (with metadata features)	**0.3088**	**0.5509**
8.	C-NGE (without metadata features)	0.3209	0.5690
9.	C-NGE (fixed latent vectors)	0.4909	0.5986

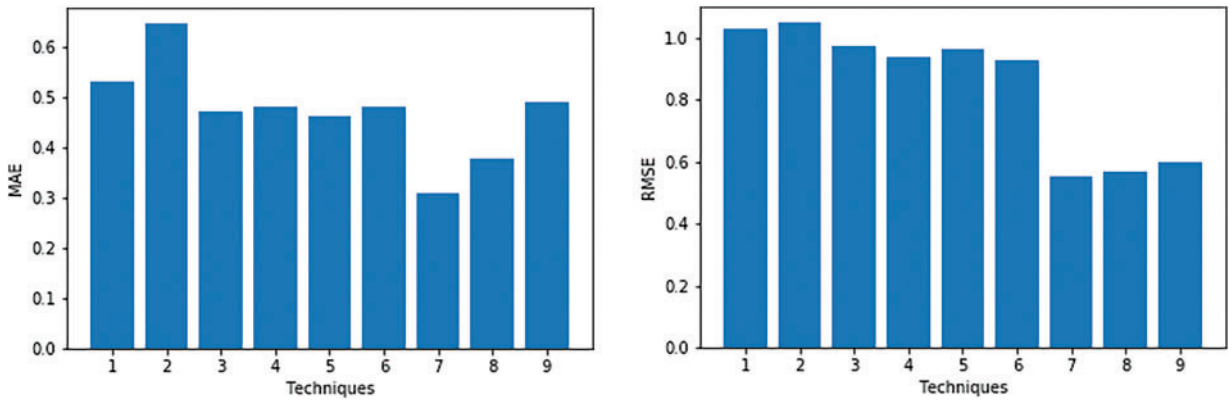


Figure 2: Comparison of MAE and RMSE for methods used in Table 4

5.6 Quantitative Findings with Error Analysis

RMSE and MAE values for three datasets employing various approaches are presented in Table 5 (Unsupervised) and Table 6 (Supervised). Note: *Improve* indicates C-NGE's relative relevance compared to the best competition. As proven, the proposed strategy consistently and significantly beats baseline methods. C-NGE outperforms NCF in MovieLens 100 K dataset with RMSE 0.858% and MAE 1.392%, but lags behind other baseline techniques.

Table 5: Unsupervised techniques

Techniques	MovieLens 100 K		Indian Regional Movies		MovieLens 1 M	
Errors	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
User-User similarity	0.7980	1.026	0.5307	1.0312	0.707	0.8810
Item-Item similarity	0.744	1.061	0.6482	1.049	0.671	0.9196
Matrix factorization	0.828	1.128	0.471	0.9713	0.6863	0.8790
Probabilistic matrix factorization	0.7564	0.9639	0.4811	0.9372	0.7241	0.9127
Blind compressed sensing	0.7356	0.9409	0.4632	0.9612	0.6917	0.8789
Matrix completion	0.8324	1.102	0.4827	0.9264	0.7196	0.9102
SVD	0.743	0.9521	0.493	0.938	0.686	0.8730
ConvMF	0.735	0.9469	0.471	0.930	0.676	0.8549

(Continued)

Table 5 (continued)

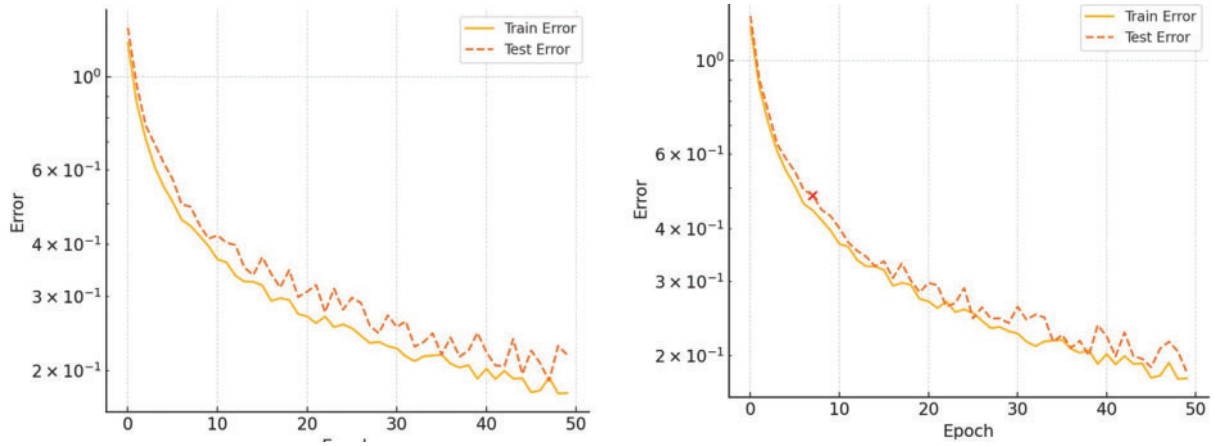
Techniques	MovieLens 100 K		Indian Regional Movies		MovieLens 1 M	
Errors	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
MLP	0.729	0.9743	0.446	0.9297	0.681	0.8773
NCF	0.718	0.9319	0.3209	0.569	0.668	0.8480
C-NGE	**0.708**	**0.9239**	**0.3088**	**0.5509**	**0.661**	**0.8421**
Improve	1.392%	0.858%	3.77%	3.181%	1.047%	0.695%

Table 6: Supervised technique

Techniques	MovieLens 100 K		Indian regional movies		MovieLens 1 M	
Errors	**MAE**	**RMSE**	**MAE**	**RMSE**	**MAE**	**RMSE**
Supervised matrix factorization	0.7173	0.9241	0.4367	0.9283	0.6659	0.8469

The C-NGE model outperforms baseline approaches on MovieLens 1 M and IRM datasets. It suggests that using user/item metadata as supplementary information can improve latent factor interpretation and rating prediction. By combining the non-linear M-NE model, the C-NGE model is expressive. MLP outperforms MF by a small margin, leaving MF a special case of MLP.

On the MovieLens 100 K and MovieLens 1 M datasets, plots of C-NGE training error versus epoch are shown in Fig. 3a and b, respectively.



(a) Plot of training and Test error versus epoch for the MovieLens 100K dataset

(b) Plot of training and Test error versus epoch for the MovieLens 1M dataset

Figure 3: Plot of model training loss on MovieLens datasets

As the C-NGE model's last hidden layer determines the model's potential, we refer it as *explanatory predictors* and evaluated the predictors of [8, 16, 32, 64]. Large predictors overfits the model and degrade the performance. Three hidden layers are used for M-NE. For example, if the explanatory predictors size is 8, the M-NE layers architecture is $32 \rightarrow 16 \rightarrow 8$, and the embedding size is 16. Firstly, it can be seen that C-NGE outperforms the state-of-the-art ConvMF and NCF methods by a substantial margin, achieving the best

performance over datasets (the relative improvement over ConvMF and NCF is 2.4% and 0.8% on RMSE, respectively, on average). For MovieLens 1M, C-NGE significantly outperforms that of ConvMF and NCF with a large predictor of 64. It reveals the high expressiveness of C-NGE by fusing non-linear M-NE model. Note that M-NE can be improved even further by adding more hidden layers (more than five hidden layers somewhat overfits the data), and here only the performance of three layers are shown.

5.7 Execution Time Analysis and Scalability

An analysis is conducted on the computing time of the C-NGE model and baseline approaches. Each solution operates on a solitary system with an NVIDIA GeForce GTX 1080 GPU. During the training phase, NCF and C-NGE require approximately 6 s per epoch on the MovieLens 100 K dataset. In contrast, on the MovieLens 1M dataset, they take around 1 min and 20 s for each epoch. By employing the early stopping criterion, it is common for all of the models to reach convergence in fewer than 25 epochs. The recommendation prediction results exhibit a relatively rapid response time during the testing phase, with a duration of only 3–4 s. MF-based techniques like PMF or SVD exhibit faster execution times than neural network-based techniques. C-NGE necessitates a comparable amount of time to NCF during the training period. However, it demands more time than SVD or PMF. During the testing phase, the suggested method demonstrates similar time efficiency to previous strategies. Hence, in practical terms, the proposed C-NGE architecture is viable for a movie recommendation system.

The issue with recommendation algorithms lies in their computational scalability, as the calculation increases exponentially with the growth in the number of users and items. Scaling the dataset negatively impacts the performance of the recommendation model, which was trained and achieved superior results on a small dataset. Consequently, effectively implementing recommendation algorithms becomes more crucial as the dataset expands. The utilization of neural networks for dimensionality reduction is employed to tackle the issues related to scalability. The total number of neuron units in each subsequent hidden layer is precisely half of the amount in the preceding hidden layer. It guarantees that each subsequent hidden layer acquires novel representations while simultaneously diminishing noise (i.e., extraneous information) from the preceding hidden layer, enabling the concentration on relevant features. Empirical studies demonstrate that employing an equal number of neuron units in each hidden layer significantly increases the cost per computation and backpropagation by about threefold. This, in turn, requires a more significant amount of data, meaning a need for more than triple the training data and training time. We obtained the following statistics in Table 7 to further investigate the average epoch needed to converge, the number of trainable parameters to learn, and the prediction time for different methods with varying dataset sizes. The average duration required to complete a single epoch is the average epoch time. Three distinct sizes of datasets are taken into account: 25%, 75%, and 100% (entire dataset). The findings are as follows: (1) As the dataset expands, there is a substantial rise in the average number of epochs required and in the number of learnable parameters and prediction time. (2) Our method necessitates fewer epochs, parameters, and prediction time during the training phase compared to the leading rival NCF method. (3) The slight disparity in performance between NCF and our approach may be attributed to the fact that NCF employs neural networks directly for the collaborative filtering task, enabling it to learn complex interactions nonlinearly. However, NCF does not consider any additional information about the users or items involved. Our approach utilizes metadata information for people and objects to model their interactions. We apply a decoder as a neural network to forecast the final rating. This method is systematic and well-defined.

Table 7: Over all datasets, average epoch, number of trainable parameters, and prediction time for different methods with varying dataset size ratios

Model	Dataset Ratio	IRM			ML100K			ML1M		
		Average Epoch (s)	Parameters (Million)	Prediction Time (s)	Average Epoch (s)	Parameters (Million)	Prediction Time (s)	Average Epoch (s)	Parameters (Million)	Prediction Time (s)
MLP	25	1	2	0.1	1	3	0.2	6	7	1
	75	1	4	0.4	2	6	0.7	11	9	3
	100	2	7	0.6	3	11	0.8	14	17	4
NCF	25	1	4	0.2	2	6	0.2	182	11	3
	75	3	11	0.6	5	15	0.6	254	19	5
	100	4	20	0.8	7	26	0.9	281	37	7
Our method (C-NGE)	25	2	2	0.1	2	3	0.2	162	5	3
	75	3	6	0.4	4	8	0.8	247	10	6
	100	5	10	0.7	6	13	1.1	278	19	7

5.8 Hyper-Parameters Tuning and Statistical Significance

Indicators that are commonly used to assess the reliability of experimental results, such as searching for hyperparameters or statistical significance, are discussed.

Keras[3] neural networks Application Programming Interface (API) is used in experiments with proposed models. The experiments employ mini-batch gradient descent with the Adaptive Moment Estimation (ADAM) optimizer. The model evaluation is conducted using learning rates of {0.0001, 0.0005, 0.001} and batch sizes of {128, 256, 512}. The M-NE model is designed with three hidden layers. To maintain fairness in comparison, all models, including C-NGE and the baseline methods, are trained under consistent hyperparameters, optimizers, and training epochs.

The proposed methods outperform other baselines with noticeable improvements, and further, one-sample paired t-tests are conducted to verify that $p < 0.02$ is statistically significant for all improvements. References [57,58] have reported most of the results on the MovieLens 1M dataset. The documented standard deviation frequently exhibits a lower value, and the disparity in documented outcomes is deemed statistically significant. The findings of the significance test should not be used as a reliable method to evaluate whether algorithm A is superior to algorithm B. The significance test does not assess the efficacy of the algorithm's configuration; instead, it examines the standard deviation within the configuration. Consequently, it is advisable only to conduct variance and significance tests if there is evidence that the algorithm being utilized is effectively employed. During the process of implementing the utilized algorithm, a significant number of errors arise. Statistical significance tests have limited utility and often result in erroneous interpretations of experimental findings.

5.9 Is Deep Learning Beneficial?

More research is needed to examine user interactions and items utilizing neural networks. It is crucial to determine whether deploying deep neural networks enhances the precision of the recommendations. We explore this by augmenting the number of hidden layers in the M-NE model. The notation M-NE-3 indicates the presence of three hidden layers, excluding the embedding layer. It has been noticed that adding additional hidden layers is advantageous for the recommendation task. The results achieved with M-NE-3 are

[3] https://keras.io/ (accessed on 25 June 2025).

auspicious, indicating that using deep neural networks for collaborative recommendation is quite efficient. The improvement is attributed to the non-linearities that arise with adding more hidden layers. Performance is degraded when the number of hidden layers is increased to six or more due to overfitting of the user-item interactions. Linear activation functions are used to verify the hidden layers. The inability to capture non-linear interactions between users and items causes the performance to degrade. The factorization strategies employed for the recommendation problem produced identical outcomes.

The performance of M-NE-0, which lacks hidden layers (consisting only of an embedding layer), diminishes, indicating that only performing an element-wise dot product between the latent vectors of the user and object is inadequate for modeling their interactions. Consequently, hidden layers are necessary to convert it.

The RMSE values are evaluated while considering different components of K. As seen in Fig. 4, the RMSE of the proposed model surpasses that of other baseline approaches. The findings are validated using the IRM and ML1M datasets. The ML100K dataset findings are omitted due to their similarity to the ML1M dataset results. The rating prediction challenge enhancements demonstrate the proposed model's superiority since deep neural networks are designed to effectively capture the underlying and interactive characteristics between users and things. When K is set to 256, all models exhibit minimum rating prediction error.

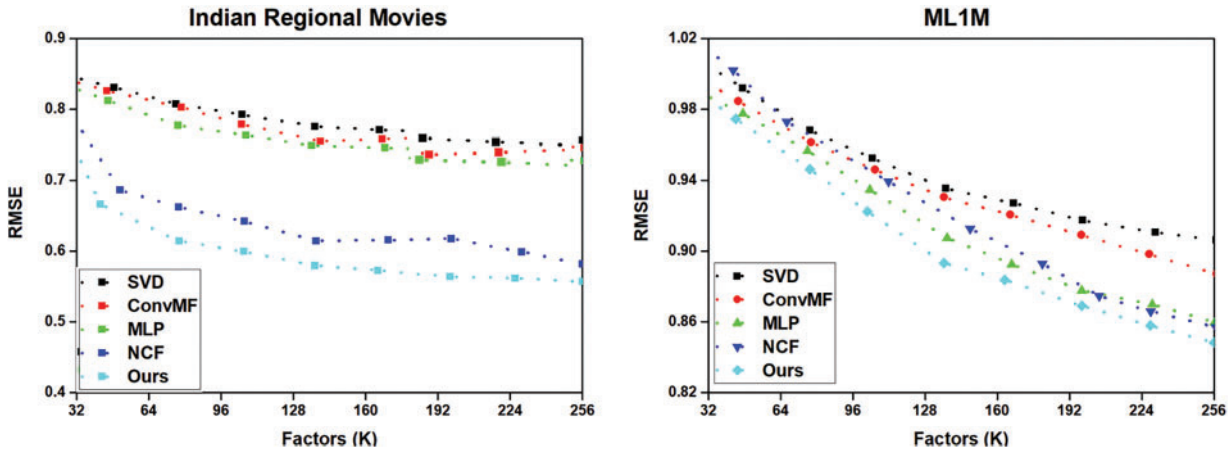


Figure 4: On the test set, RMSE performance comparison with different embedding size *K*

The root mean square error (RMSE) validation error associated with the dropout ratios is displayed in Fig. 5. Dropouts are a regularization technique to mitigate overfitting, with values between 0.1 and 0.9. As the rate of students leaving school without completing their education increases, the inaccuracy in the validation process varies. This could be attributed to a higher rate of dropouts, which may result in underfitting the model. The proposed model underwent testing using four distinct activation functions. The results revealed that ReLU exhibited the lowest error rates when smaller dropouts were applied to the IRM dataset, whereas sigmoid yielded the lowest error rates for the ML1M dataset. The model, which incorporates dropouts over 0.7, exhibits an inability to forecast ratings reliably for both datasets, leading to the underfitting of the model. As dropouts increase, the identity function exhibits almost linear behavior, albeit with a significant validation error.

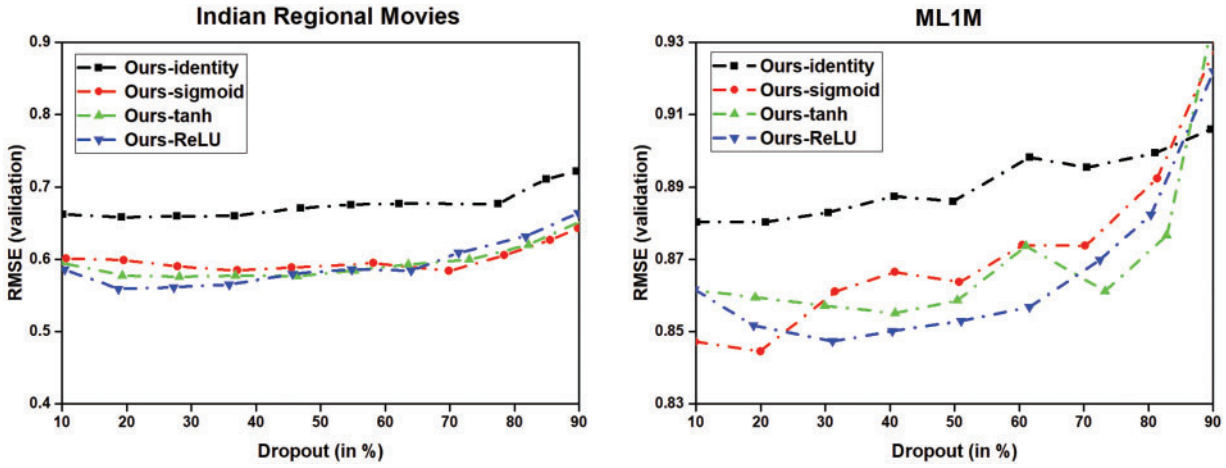


Figure 5: On the first hidden layer, RMSE validation error w.r.t. different activation functions and dropout ratios

6 Conclusion

In this paper, we have proposed a deep Collaborative Neural Generative Embedding (C-NGE) model that generates the latent vectors dynamically and the final rating using the reparameterization trick. The C-NGE model deals with uncertainty in the ratings without retraining the model when a cold user or cold item arrives. The implementation and evaluation of C-NGE and other baseline methods on Indian Regional Movies (IRM) and MovieLens datasets are discussed. The proposed work demonstrates the integration of metadata features as sampled noise and rating information, using a unified neural networks framework. It was observed that the C-NGE model gave a better performance in comparison with the baseline models due to the incorporation of metadata features. The proposed model can further be improved by augmenting the dataset with other auxiliary information, such as user reviews and multimodal data (like still images and visual semantics), to get a better idea of the user's preference. A limitation of the C-NGE model is its reliance on the quality of metadata features; poor or incomplete metadata can adversely impact its performance.

Acknowledgement: The authors sincerely thank the anonymous reviewers for their insightful comments and constructive suggestions, which have significantly improved the quality and clarity of this manuscript.

Funding Statement: This research received no external funding.

Author Contributions: The authors confirm contribution to the paper as follows: Study conception and design: Ravi Nahta, Nagaraj Naik. Methodology: Ravi Nahta, Nagaraj Naik. Implementation: Ravi Nahta, Nagaraj Naik, Srivinay, Swetha Parvatha Reddy Chandrasekhara. Validation: Ravi Nahta, Nagaraj Naik, Srivinay, Swetha Parvatha Reddy Chandrasekhara. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are openly available at: https://grouplens.org/datasets/movielens/100K/ (accessed on 25 June 2025). https://grouplens.org/datasets/movielens/1m/ (accessed on 25 June 2025). https://github.com/aitnagaraj/data (accessed on 25 June 2025).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Communication Crafts & MICA–The School of Ideas. Indian OTT Platforms Report 2019 (Tech. Rep.). Ahmedabad, India. [cited 2025 Mar 16]. Available from: https://communicationcrafts.in/wp-content/uploads/2019/12/Indian_ott_report2019.pdf.
2. Gope J, Jain SK. A survey on solving cold start problem in recommender systems. In: 2017 International Conference on Computing, Communication and Automation (ICCCA); 2017 May 5–6; Greater Noida, India. p. 133–8.
3. Nahta R, Chauhan GS, Meena YK, Gopalani D. CF-MGAN: collaborative filtering with metadata-aware generative adversarial networks for top-N recommendation. Inf Sci. 2025;689(2):121337. doi:10.1016/j.ins.2024.121337.
4. Zhang S, Yao L, Sun A, Tay Y. Deep learning based recommender system: a survey and new perspectives. ACM Comput Surv. 2019;52(1):1–38. doi:10.1145/3285029.
5. Kingma DP, Welling M. Auto-encoding variational Bayes. arXiv:1312.6114. 2013.
6. Li X, She J. Collaborative variational autoencoder for recommender systems. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2017 Aug 13–17; Halifax, NS, Canada. p. 305–14.
7. Gheewala S, Xu S, Yeom S. In-depth survey: deep learning in recommender systems—exploring prediction and ranking models, datasets, feature analysis, and emerging trends. Neural Comput Appl. 2025;37(17):10875–947. doi:10.1007/s00521-024-10866-z.
8. Shahab M, Hassim YMM, Ghazali R, Javid I, Arbaiy N. An in-depth strategy using deep generative adversarial networks for addressing the cold start in movie recommendation systems. In: International Conference on Soft Computing and Data Mining; 2024; Cham, Switzerland: Springer. p. 136–43.
9. Siet S, Peng S, Ilkhomjon S, Kang M, Park DS. Enhancing sequence movie recommendation system using deep learning and KMeans. Appl Sci. 2024;14(6):2505. doi:10.3390/app14062505.
10. Padmavathi A, Amrutha G, Sah RK, Chapagain B, Manasa A. Performance evaluation of movie-based recommendation systems using hybrid machine learning models. In: 2024 5th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI); 2024 Jan 18–19; Lalitpur, Nepal. p. 195–201.
11. Anwar K, Zafar A, Iqbal A. An efficient approach for improving the predictive accuracy of multi-criteria recommender system. Int J Inf Technol. 2024;16(2):809–16. doi:10.1007/s41870-023-01547-6.
12. Mao Y, Woradit K, Cosh K. Hybrid movie recommendation system with user partitioning and log likelihood content comparison. IEEE Access. 2025;13(7):11609–22. doi:10.1109/access.2025.3529515.
13. Sinha BB, Sinha R, Priye V. Beyond classical approaches: redefining the landscape of high-accurate movie recommendation using QNN. J Supercomput. 2025;81(1):347. doi:10.1007/s11227-024-06746-x.
14. Kula M. Metadata embeddings for user and item cold-start recommendations. arXiv:1507.08439. 2015.
15. Vasile F, Smirnova E, Conneau A. Meta-Prod2Vec: product embeddings using side-information for recommendation. In: Proceedings of the 10th ACM Conference on Recommender Systems; 2016 Sep 15–19; Boston, MA, USA. p. 225–32.
16. Fernández-Tobías I, Cantador I, Tomeo P, Anelli VW, Noia T. Addressing the user cold start with cross-domain collaborative filtering: exploiting item metadata in matrix factorization. User Model User-Adapt Interact. 2019;29(2):443–86. doi:10.1007/s11257-018-9217-6.
17. Soares M, Viana P. Tuning metadata for better movie content-based recommendation systems. Multimed Tools Appl. 2015;74(17):7015–36. doi:10.1007/s11042-014-1950-1.
18. Yoon YC, Lee JW. Movie recommendation using metadata based Word2Vec algorithm. In: 2018 International Conference on Platform Technology and Service (PlatCon); 2018 Jan 29–31; Jeju, Republic of Korea. p. 1–6.
19. Henk V, Vahdati S, Nayyeri M, Ali M, Yazdi HS, Lehmann J. Metaresearch recommendations using knowledge graph embeddings. In: The AAAI-19 Workshop on Recommender Systems and Natural Language Processing (RecNLP); 2019 Jan 27–28; Honolulu, HI, USA.
20. D'Addio RM, Marinho RS, M. MG. Combining different metadata views for better recommendation accuracy. Inf Systs. 2019;83(2):1–12. doi:10.1016/j.is.2019.01.008.

21. Zhao L, Lu Z, Pan SJ, Yang Q, Xu W. Matrix factorization+ for movie recommendation. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16); 2016 Jul 9–15; New York, NY, USA. p. 3945–51.
22. Krishnan A, Sharma A, Sankar A, Sundaram H. An adversarial approach to improve long-tail performance in neural collaborative filtering. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management; 2018 Oct 22–26; Torino, Italy. p. 1491–4.
23. Yuan F, Yao L, Benatallah B. Adversarial collaborative auto-encoder for top-n recommendation. In: 2019 International Joint Conference on Neural Networks (IJCNN); 2019 Jul 14–19; Budapest, Hungary: IEEE. p. 1–8.
24. Dai Q, Li Q, Tang J, Wang D. Adversarial network embedding. In: Proceedings of the AAAI Conference on Artificial Intelligence; 2018 Feb 2–7; New Orleans, LA, USA. p. 2167–74.
25. Bharadhwaj H, Park H, Lim BY. RecGAN: recurrent generative adversarial networks for recommendation systems. In: Proceedings of the 12th ACM Conference on Recommender Systems; 2018 Oct 2; Vancouver, BC, Canada. p. 372–6.
26. Wang S, Tang J, Wang Y, Liu H. Exploring implicit hierarchical structures for recommender systems. In: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015); 2015 Jul 25–31; Buenos Aires, Argentina. p. 1813–9.
27. Yi B, Shen X, Liu H, Zhang Z, Zhang W, Liu S, et al. Deep matrix factorization with implicit feedback embedding for recommendation system. IEEE Trans Ind Inform. 2019;15(8):4591–601. doi:10.1109/tii.2019.2893714.
28. Chen J, Zhuang F, Hong X, Ao X, Xie X, He Q. Attention-driven factor model for explainable personalized recommendation. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval; 2018 Jul 8–12; Ann Arbor, MI, USA. p. 909–12.
29. Chae DK, Kang JS, Kim SW, Lee JT. CFGAN: a generic collaborative filtering framework based on generative adversarial networks. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management; 2018 Oct 22–26; Torino, Italy. p. 137–46.
30. Hu GN, Dai XY, Qiu FY, Xia R, Li T, Huang SJ, et al. Collaborative filtering with topic and social latent factors incorporating implicit feedback. ACM Trans Knowl Discov Data. 2018;12(2):1–30. doi:10.1145/3127873.
31. Zhao W, Wang B, Yang M, Ye J, Zhao Z, Chen X, et al. Leveraging long and short-term information in content-aware movie recommendation via adversarial training. IEEE Trans Cybern. 2020;50(11):4680–93. doi:10.1109/tcyb.2019.2896766.
32. Cheng Z, Chang X, Zhu L, Kanjirathinkal RC, Kankanhalli M. MMALFM: explainable recommendation by leveraging reviews and images. ACM Trans Inf Syst. 2019;37(2):1–28.
33. Cheng Z, Ding Y, Zhu L, Kankanhalli M. Aspect-aware latent factor model: rating prediction with ratings and reviews. In: Proceedings of the 2018 World Wide Web Conference; 2018 Apr 23–27; Lyon, France. p. 639–48.
34. He X, Liao L, Zhang H, Nie L, Hu X, Chua TS. Neural collaborative filtering. In: Proceedings of the 26th International Conference on World Wide Web; 2017 Apr 3–7; Perth, WA, Australia. p. 173–82.
35. Zhuang F, Zhang Z, Qian M, Shi C, Xie X, He Q. Representation learning via dual-autoencoder for recommendation. Neural Networks. 2017;90(5):83–9. doi:10.1016/j.neunet.2017.03.009.
36. Wang C, Blei DM. Collaborative topic modeling for recommending scientific articles. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2011 Aug 21–24; San Diego, CA, USA. p. 448–56.
37. Dong X, Yu L, Wu Z, Sun Y, Yuan L, Zhang F. A hybrid collaborative filtering model with deep structure for recommender systems. In: Proceedings of the AAAI Conference on Artificial Intelligence; 2017 Feb 4–9; San Francisco, CA, USA. p. 1309–15.
38. Nahta R, Meena YK, Gopalani D, Chauhan GS. Embedding metadata using deep collaborative filtering to address the cold start problem for the rating prediction task. Multimed Tools Appl. 2021;80(12):18553–81. doi:10.1007/s11042-021-10529-4.
39. Agarwal P, Verma R, Majumdar A. Indian regional movie dataset for recommender systems. arXiv:1801.02203. 2018.

40. Khan Z, Khan Z, Iltaf N. ConvSeq-MF: convo-sequential matrix factorization for recommender system. Neurocomputing. 2025;618(4):128932. doi:10.1016/j.neucom.2024.128932.
41. Gogna A, Majumdar A. Blind compressive sensing framework for collaborative filtering. arXiv:1505.01621. 2015.
42. Gleichman S, Eldar YC. Blind compressed sensing. IEEE Trans Inf Theory. 2011;57(10):6958–75. doi:10.1109/tit.2011.2165821.
43. Taromi AD, Heydari S, Hooshmand M, Ramezani M. RSAttAE: an information-aware attention-based autoencoder recommender system. arXiv:2502.06705. 2025.
44. Sarwar B, Karypis G, Konstan J, Riedl J. Incremental singular value decomposition algorithms for highly scalable recommender systems. In: Fifth International Conference on Computer and Information Science; 2002 Sep 30–Oct 2; Sao Paulo, Brazil. p. 27–8.
45. Khatter H, Singh P, Ahlawat A, Shrivastava AK. Two-tier enhanced hybrid deep learning-based collaborative filtering recommendation system for online reviews. Comput Intell. 2025;41(3):e70062. doi:10.1111/coin.70062.
46. Jain A, Jain G, Nagar S, Singh PK, Dhar J. Rating distribution-aware deep cognitive convolution matrix factorization for recommendation systems. Arab J Sci Eng. 2025;50(10):7441–62. doi:10.1007/s13369-024-09361-3.
47. Elahi E, Anwar S, Al-kfairy M, Rodrigues JJ, Ngueilbaye A, Halim Z, et al. Graph attention-based neural collaborative filtering for item-specific recommendation system using knowledge graph. Expert Syst Appl. 2025;266(1):126133. doi:10.1016/j.eswa.2024.126133.
48. Yan Y, Moreau C, Wang Z, Fan W, Fu C. Transforming movie recommendations with advanced machine learning: a study of NMF, SVD, and K-means clustering. In: 2024 4th International Symposium on Computer Technology and Information Science (ISCTIS); 2024 Jul 12–14; Xi'an, China. p. 178–81.
49. Ouyang Z, Zhang C, Jia Y, Vosoughi S. Scaled supervision is an implicit Lipschitz regularizer. In: Proceedings of the International AAAI Conference on Web and Social Media; 2025 Jun 23–26; Copenhagen, Denmark. p. 1419–35.
50. Shaikh S, Kagita VR, Kumar V, Pujari AK. Data augmentation and refinement for recommender system: a semi-supervised approach using maximum margin matrix factorization. Expert Syst Appl. 2024;238(6):121967. doi:10.1016/j.eswa.2023.121967.
51. Deng Y, Zhou W, Haq AU, Ahmad S, Tabassum A. Differentially private recommender framework with dual semi-autoencoder. Expert Syst Appl. 2025;260(4):125447. doi:10.1016/j.eswa.2024.125447.
52. Sun Y, Liu Q. Collaborative filtering recommendation based on K-nearest neighbor and non-negative matrix factorization algorithm. J Supercomput. 2025;81(1):79. doi:10.1007/s11227-024-06537-4.
53. Gunawardana A, Shani GA. survey of accuracy evaluation metrics of recommendation tasks. J Mach Learn Res. 2009;10:2935–62.
54. Ma H, Yang H, Lyu MR, King I. SorRec: social recommendation using probabilistic matrix factorization. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management; 2008 Oct 26–30; Napa Valley, CA, USA. p. 931–40.
55. Singh AP, Gordon GJ. Relational learning via collective matrix factorization. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining; 2008 Aug 24-27; Las Vegas, NV, USA. p. 650–8.
56. Srebro N, Rennie J, Jaakkola T. Maximum-margin matrix factorization. In: Advances in neural information processing systems. Vancouver, BC, Canada: MIT Press; 2004. 17 p.
57. Li D, Chen C, Liu W, Lu T, Gu N, Chu SM. Mixture-rank matrix approximation for collaborative filtering. In: Guyon I, Luxburg UV, Bengio S, Bengio S, editors. Advances in neural information processing systems. Vol. 30. Red Hook, NY, USA: Curran Associates Inc.; 2017. p. 477–85.
58. Strub F, Gaudel R, Mary J. Hybrid recommender system based on autoencoders. In: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems; 2016 Sep 15; New York, NY, USA: ACM. p. 11–16.