





Enhancing Movie Recommendations: An Ensemble-Based Deep Collaborative Filtering Approach Utilizing AdaMVRGO Optimization

V Lakshmi Chetana^{1*}, Hari Seetha²

¹ School of Computer Science and Engineering, VIT-AP University, Amaravati 522237, Andhra Pradesh, India

² Center of Excellence, AI and Robotics, VIT-AP University, Amaravati 522237, Andhra Pradesh, India

Corresponding Author Email: seetha.hari@vitap.ac.in

Copyright: ©2023 IETA. This article is published by IETA and is licensed under the CC BY 4.0 license (<http://creativecommons.org/licenses/by/4.0/>).

<https://doi.org/10.18280/ts.400602>

ABSTRACT

Received: 20 March 2023

Revised: 31 July 2023

Accepted: 1 November 2023

Available online: 30 December 2023

Keywords:

adaptive moment variance reduced gradient optimization, matrix factorization, movie recommendation, ensemble deep neural networks and recommendation systems

Collaborative filtering, while a powerful tool in movie recommendation systems, encounters substantial challenges such as sparsity, scalability, diversity, and interpretability, which influence recommendation fidelity. Although the sparsity issue can be tackled through traditional model-based collaborative filtering algorithms like matrix factorization, these models often fail to capture the full depth of user-movie interactions due to their reliance on a simple dot product for rating prediction. Recently, research has leaned towards the application of deep learning to harness the complex and non-linear relationships between users and movies. However, these deep learning methods, despite their non-linear attributes, are susceptible to high variance and overfitting, potentially compromising their capacity for generalization. In the present study, an ensemble of neural networks has been implemented to diminish variance and generalization error by amalgamating the predictions from multiple models. This approach enhances overall performance and tackles the inherent limitations of deep learning approaches. A novel ensemble-based deep collaborative filtering model (Deep CF), in concert with a unique optimizer (AdaMVRGO), has been introduced to address sparsity and to exploit the non-linear, complex relationships between users and movies. It also aims to minimize the variance and generalization error of the neural network. The proposed architecture, termed Deep CF-AdaMVRGO, employs an ensemble of multi-layer perceptrons (MLP) to augment prediction accuracy. A pioneering optimizer, the adaptive moment variance reduced gradient optimization (AdaMVRGO), has been developed, drawing upon the ADAM and SVRG optimizers. This optimizer eliminates noise by calculating the first and second moments of predicted ratings using a variance-reduced gradient, similar to the SVRG algorithm, thereby expediting algorithm convergence. It is employed to fine-tune the parameters of the MLP and decrease the reconstruction error. Deep CF-AdaMVRGO has been evaluated against six existing models using the RMSE metric on the M-1M and M-10M datasets. The simulation results demonstrated that the proposed framework outperformed state-of-the-art deep learning-based collaborative filtering approaches on both datasets in terms of lower RMSE values. Further, the performance of the proposed AdaMVRGO optimizer within the ensemble framework was compared to existing optimizers such as Adagrad, RMSProp, ADAM, and SVRG on M-100K, M-1M, and M-10M datasets using RMSE, MAE, and MSE metrics. Experimental results affirmed that AdaMVRGO converged more rapidly to the optimum compared to other optimizers.

1. INTRODUCTION

Over several decades, recommendation systems have emerged as a promising research field, harnessing user preferences to curate effective recommendations [1-3]. The algorithms underpinning these systems are widely employed across diverse domains ranging from movie and music recommendations to product suggestions, crop selection, and e-book recommendations [4]. The primary objective of a recommendation system is to predict users' preferences for movies, music, products, crops, books, and news based on their historical preferences [5-8].

Broadly, recommendation systems can be categorised into

three primary types: collaborative filtering, content-based filtering, and hybrid filtering. Among these, collaborative filtering has been acknowledged as the most effective and widely utilised method for generating recommendations [9]. This technique predicts user ratings by leveraging historical ratings and ratings from similar users [10-12]. Collaborative filtering can be further divided into two subcategories: memory-based and model-based methods [13-15]. The former, also termed a "neighbourhood-based approach," employs statistical techniques to identify similar users and items, with unknown ratings predicted based on these similarities. The latter, on the other hand, uses machine learning techniques to learn the rating matrix and predict unknown ratings. However,

collaborative filtering techniques face significant challenges, including data sparsity, scalability, cold start, and shilling attacks [16], all of which can negatively impact the accuracy and performance of recommendations.

Data sparsity, referring to the paucity of data in the rating matrix due to unknown ratings, is a particular challenge. Matrix factorization, an efficient model-based collaborative filtering strategy, has been used to tackle this problem [17]. It decomposes the rating matrix into a pair of low-dimensional rectangular matrices and predicts unknown ratings using a simple dot product [11, 15, 18]. However, this linear dot product may not fully encapsulate the complex interactions between users and movies. Hence, current research efforts are directed towards the application of deep learning algorithms to address the limitations of traditional recommendation systems [19]. Deep learning-based latent factor models are adept at capturing the hidden, non-linear, and crucial interactions between users and movies, thereby enhancing prediction accuracy [17].

Ensemble learning is a powerful strategy renowned for enhancing the accuracy, resilience, and diversity of learning algorithms. Recently, recommendation systems that leverage ensemble learning have gained considerable attention [20]. Deep ensemble models amalgamate the benefits of both deep and ensemble learning, thereby improving the overall performance of recommendation systems [20, 21]. Ensemble-based deep collaborative filtering holds several advantages over deep collaborative filtering, such as: 1) superior performance over individual models, particularly in the presence of noisy data; 2) the ability to generate more diverse recommendations compared to individual models; and 3) robustness against overfitting by virtue of reduced model variance.

In this paper, we propose a deep collaborative filtering framework, termed Deep CF-AdaMVRGO, for movie recommendation systems to effectively handle sparsity issues [22]. This framework is underpinned by an optimizer known as the Adaptive Moment Variance Reduced Gradient Optimizer, as proposed in the study [11]. The deep collaborative filtering framework employs an ensemble of multilayer perceptrons to augment prediction performance. Each sub-model is fine-tuned using the proposed AdaMVRGO optimizer to minimize the reconstruction error. The final predicted rating is obtained by averaging the predicted values derived from the sub-models within the ensemble [11].

Typically, optimization algorithms like SGD, Adagrad, RMSPROP, ADAM, and SVRG are harnessed for parameter tuning. While ADAM is often considered the preferred optimization algorithm in deep learning applications, it is known to suffer from slow convergence and generalization issues [23]. Convergence problems persist with ADAM due to high variance. The proposed AdaMVRGO algorithm is conceived by combining the strengths of the ADAM and SVRG optimizers [11]. One key advantage of the proposed algorithm is that it manages to reduce variance during the gradient calculation itself. The first and second moments are computed based on these variance-reduced gradients. The proposed framework, when trained with AdaMVRGO, demonstrated superior performance compared to Adagrad, RMSPROP, ADAM, and SVRG.

In the experimental section, the proposed model was appraised using metrics such as mean square error (MSE), mean absolute error (MAE), and root mean square error

(RMSE) on the M-100K, M-1M, and M-10M datasets [11]. Additionally, the proposed model was benchmarked against state-of-the-art deep learning-based collaborative filtering techniques [24-29] using the M-1M and M-10M datasets [30].

The remainder of this paper is organized as follows: Section 2 presents a literature review on "movie recommendation systems". Section 3 discusses the prerequisite groundwork for the proposed framework. The proposed framework and its pseudocode are presented in Section 4. Section 5 provides a comprehensive discussion of the findings and interpretation of the results. Finally, Section 6 concludes the paper and outlines potential future enhancements.

2. LITERATURE STUDY

Huang et al. [31] introduced a technique known as Neural Embedding Collaborative Filtering (NECF) matrix factorization. This method utilizes a probabilistic autoencoder to generate neural embedding vectors from user-item input. These vectors are subsequently used to represent the user's hidden features via a regression equation employing single-point negative sampling [24, 31]. The method was tested on the M-1M and Pinterest datasets, and the results revealed that this strategy outperformed its baseline counterparts. However, it still grapples with the issue of data sparsity.

Sun et al. [24] developed an innovative framework that amalgamates plot texts and movie ratings to enhance prediction accuracy. They designed and tested a deep plot-aware generalized matrix factorization on the MovieLens datasets [24, 31]. Even though the integration of additional knowledge improved the model, it led to increased computational complexity.

Feng et al. [32] proposed a unique similarity approach that considers both linear and non-linear correlations. To bolster prediction accuracy and resilience, this method combined multifactor similarity with global rating information. However, the approach was found lacking in terms of scalability.

Taking into account the historical data of users and items, Fu et al. [25] devised a novel deep learning technique for movie recommendations. Initially, user and item vectors were trained to incorporate semantic information, reflecting the relationships between items and users [11]. A multi-view feed-forward neural network was then employed to predict ratings from these embedded vectors. The proposed model was evaluated on the MovieLens 1M and MovieLens 10M datasets [25]. Despite its promising performance, the model was computationally intensive and encountered issues with generalization.

Xue et al. [33] proposed an item-based collaborative filtering model using deep learning, dubbed Deep-ICF, to decipher the nonlinear and intricate relationships between items. Deep-ICF employs a simple average to calculate the predicted rating. An extension to this model, termed Deep-ICF+a, employs adaptive pooling with attention to discern higher-order interactions between items. While promising, these models are computationally demanding and are more susceptible to overfitting.

It is noteworthy that previous works often necessitate extended computational time to converge and frequently encounter generalization issues. More recent research has concentrated on developing sophisticated models that incorporate deep learning techniques. These models utilize neural networks to capture complex user-item relationships

and accommodate a better representation of intricate interactions. Moreover, optimization algorithms such as stochastic gradient descent have been deployed to diminish computational time and promote convergence. These advancements have significantly ameliorated the accuracy and scalability of recommendation systems, thereby addressing the limitations of prior works.

The key contributions of this paper are as follows:

1. We have established a deep collaborative filtering framework to capture the hidden, nonlinear, and pivotal relationships between users and movies.
2. The framework employs an ensemble of multilayer perceptrons to train the rating matrix and predict unknown ratings. This ensemble improves generalization by training each MLP within the ensemble with a unique set of latent factors.
3. Each sub-model within the ensemble is trained using a novel optimizer known as AdaMVRGO, with the aim to reduce the reconstruction error.
4. Experiments were conducted on the M-100K, M-1M, and M-10M datasets. The results demonstrate that the proposed framework outshines existing methods.

3. PRELIMINARIES

3.1 Data representation

Let ' m ', ' n ' represent the number of users and movies in a movie recommendation system, respectively [34, 35], so $m \times n$ represents the rating matrix's size, which is defined as $R = \left\{ \frac{r_{ij}}{i \in m, j \in n \text{ and } 1 \leq i \leq m, 1 \leq j \leq n} \right\}$, ratings given by the user for a movie [6]. For our work, we used explicit ratings, i.e., ratings given explicitly by the user. An explicit rating is quantifiable feedback given by the user for a specific movie. It determines the extent to which a user prefers the movie. Most of the time, users may not be interested in giving their feedback explicitly, resulting in a sparse rating matrix. This data sparsity hinders the recommendation system's precision and performance [35].

3.2 Matrix factorization

It is a latent factor model to address sparsity issue [36, 37]. It predicts the unknown ratings by splitting the rating matrix 'R' into two latent matrices called user matrix 'P' and movie matrix 'Q' of order ' $m \times k$ ' and ' $n \times k$ ', respectively [38]. The user matrix 'P' depicts the associations across the users and ' k ' latent features. Similarly, the movie matrix 'Q' depicts the associations between the movies and the ' k ' latent features. Eqs. (1) and (2) indicate the rating matrix 'R' as well as the latent matrices 'P' and 'Q' [39]. The predicted rating matrix is generated using Eq. (3).

$$R: U_m \rightarrow M_n \quad (1)$$

$$P: U_m \rightarrow L_k \text{ and } Q: M_n \rightarrow L_k \quad (2)$$

$$\hat{R} = P \times Q^T \cong R \quad (3)$$

The dot product of user and movie vectors, as shown in Eq. (4) predicts the unknown rating.

$$\hat{r}_{ij} = p_i * q_j^T \quad (4)$$

where, \hat{r}_{ij} is the predicted rating of user ' i ' in the user vector p_i and movie ' j ' in the movie vector q_j [24]. Eq. (5) shows the deviation between the actual and expected ratings [40].

$$\min_{p_i, q_j} \sum_{i, j \in U, V} (r_{ij} - \hat{r}_{ij}) \quad (5)$$

To avoid overfitting, the model is redefined in Eq. (6).

$$\min_{p_i, q_j} \sum_{i, j \in U, V} (r_{ij} - \hat{r}_{ij}) + \gamma (\|p_i\|^2 + \|q_j\|^2) \quad (6)$$

where, γ is a regularisation parameter, $\|p_i\|^2$ and $\|q_j\|^2$ are the l2-norms of the user vector p_i and the movie vector q_j , respectively [41]. However, the problem with traditional matrix factorization is that the dot product is linear, and it cannot capture user-movie interactions completely. Many studies have shown that deep neural networks, rather than a simple dot product, can capture user-item interactions nonlinearly well.

3.3 Neural network ensemble

Ensemble learning is the process of combining the different models' predictions to enhance their accuracy [42, 43]. Neural networks are non-linear, can learn complex patterns in the data, but suffer from high variance. Ensemble modelling of neural networks helps to reduce the variance by building different models as opposed to just one and combining their predictions to enhance the overall performance. The benefits of this ensemble are: First, it can help reduce the variance of the predictions, thereby improving the model's generalisation performance [44]. Second, it helps to increase the precision of the predictions by combining the strengths of multiple neural networks [45]. Third, it can help the model handle noise and errors in the data better. The various ways to combine the predictions in ensemble learning in movie recommendations include:

- Voting: This is the simplest method, and it simply takes the most common prediction among the individual models. For example, if three models predict that a user will like a movie and two models predict that the user will not, then the ensemble model will predict that the user will like the movie.
 - Averaging: This method takes the average of the individual predictions. This helps to reduce the variance in predictions, thereby improving the model's generalisation performance [44].
 - Weighted averaging: This method weights the individual predictions according to their accuracy. It helps enhance prediction accuracy by assigning more weight to the more accurate models.
 - Stacking: This method uses a meta-model approach to make the final predictions. The meta-model uses the predictions of each model to make final predictions. This helps improve prediction accuracy by combining the strengths of each model [46].
- The selection of ensemble methods is application-dependent. If the objective is to improve prediction accuracy, for instance, then a voting or weighted averaging scheme may be a better choice. However, if the objective is to develop a model more robust to noise, then an averaging scheme may be

a better choice [6].

$$\omega_{t+1} = \omega_t - \frac{\delta}{\sqrt{\hat{v}_t + \epsilon}} \hat{u}_t \quad (11)$$

3.4 Optimization

The optimization algorithm has become essential for training a deep learning architecture. It starts with defining the loss function and ends with minimizing it using any gradient optimizer. Gradient-based optimization algorithms are extensively employed in latent factor-based collaborative filtering algorithms to learn explicit rating-based recommendation models [18, 47]. There are various optimizers to serve this purpose. The major problem with gradient-based optimisation algorithms is determining the learning rate, as model convergence depends on the learning rate. Lower learning rates need more time for convergence, while higher learning rates may skip the optimal solution.

The ADAM optimizer is a widely used stochastic optimization algorithm for deep learning models [23]. Even though it is considered the best adaptive learning optimization algorithm, it still suffers from convergence problems due to its inherent variance. Variance makes convergence harder, especially when parameters are close to their optimal values. According to conventional research, a declining learning rate can help reduce variance [48, 49]. When the learning rate is low, however, the training loss converges slowly [50]. We use a stochastic variance-reduced gradient in this paper to reduce variance during the ADAM process. This paper involves generating AdaMVRGO by reducing the variance of ADAM using SVRG optimizers to learn the recommendation model.

ADAM [51], a variant of the stochastic gradient algorithm [52]. ADAM combines the benefits of the AdaGrad [53] and RMSProp [54] algorithms. ADAM works well with both sparse and noisy data. The first moment uses an exponential weighted average of past gradients to converge to the minima faster, which is given in Eq. (7).

$$u_t = \beta_1 u_{t-1} + (1 - \beta_1) g_t \quad (7)$$

The second moment, which is given in Eq. (8), employs exponential moving averages (i.e., the cumulative sum of the gradients, uncentered variance).

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) * g_t^2 \quad (8)$$

where, u_t : sum of the gradients at time 't', u_{t-1} : sum of the gradients at time $t-1$, g_t^2 : sum of squares of gradients at time t, v_{t-1} : sum of squares of gradients at time $t-1$, g_t : gradient of the loss function L w.r.t, ω , δ_t : learning rate at time t, β_1 and β_2 are the moving average parameters (0.9), ω_{t+1} represents weights at time $t+1$ and ω_t represents weights at time t [55].

As the first and second moment vectors are initialized to zero in the algorithm [56], it is observed that the algorithm is biased towards zero. Hence, these vectors are corrected, as shown in Eqs. (9) and (10).

$$\hat{u}_t = \frac{u_t}{1 - \beta_1^t} \quad (9)$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \quad (10)$$

Finally, the ADAM update rule is given as shown in Eq. (11).

Although ADAM is considered the best optimization algorithm for deep learning applications, it still suffers from slow convergence and generalization issues. Wilson et al. [57] proved that adaptive algorithms are less generalizable than SGD [58]. Liu et al. [23] used adaptive gradient methods as opposed to nonadaptive gradient methods to address the issues of poor convergence for specific objective functions, no advantage from utilising moving averages, and the lowest generalisation performance. This may be due to the presence of large variance in the early epochs of training a model. Lower variance improves the convergence rate without any change to the learning rate [59]. Variance reduction methods are used to reduce the variance and achieve better generalization and a fast convergence rate.

4. METHODOLOGY

4.1 Proposed framework

As matrix factorization cannot completely capture the latent features, we employed neural networks to predict the sparse ratings [60]. Neural networks are usually non-linear and can capture trivial or complex relationships between the user and movies. But neural networks suffer from high variance. So, in this paper, an ensemble of multilayer perceptrons is used for a movie recommendation system. An ensemble of neural networks is considered to reduce variance and generalization error. The proposed Deep-CF framework is presented in Figure 1.

Deep collaborative filtering (Deep CF) uses an ensemble of multilayer perceptrons to train the model by capturing the nonlinear interactions between the user and movies. The proposed framework uses three multi-layer perceptrons (MLP): - M1, M2 and M3. M1 uses one hidden layer, M2 uses two hidden layers, and M3 uses three hidden layers with dropout. The predicted rating from each MLP is optimized using a novel optimizer called AdaMVRGO to reduce the reconstruction error. The novel optimizer cuts down on noise by figuring out the first and second moments using the SVRG algorithm's variance-reduced gradient. This leads to fast convergence. The final rating is obtained from averaging the predicted ratings of the three models (M1, M2, and M3). The structure of an MLP used in the ensemble framework is presented in Figure 2.

In this layer, the rating matrix is split into user and movie embeddings, which are considered latent features embeddings [11]. In the merge/concatenation layer, the latent features are given to a simple dot product [11]. This result is sent to an ensemble of MLPs to anticipate the sparse rating. The formula for predicting the rating [7] of each MLP is shown in Eq. (12).

$$\hat{r}_{ij} = f(P, Q^T, \omega_f) \quad (12)$$

where, $P \in R^{m \times k}$ and $Q \in R^{n \times k}$ represent the latent matrices of users and movies, respectively [61]. \hat{r}_{ij} represent the predicted score given by the user p_i and the movie q_j . ω_f indicate the parameters of the function f. The objective functions for M1, M2, and M3 are given in Eqs. (13), (14), and (15).

$$f_{M_1}(P, Q^T) = \phi_{out}(\phi_{h_1}(\phi_{in}(P, Q^T, \omega_{in}), \omega_{h_1}), \omega_{out}) \quad (13)$$

$$f_{M_2}(P, Q^T) = \phi_{out}(\phi_{h_2}(\phi_{h_1}(\phi_{in}(P, Q^T, \omega_{in}), \omega_{h_1}), \omega_{h_2}), \omega_{out}) \quad (14)$$

$$f_{M_3}(P, Q^T) = \phi_{out}(\phi_{h_3}(\phi_{h_2}(\phi_{h_1}(\phi_{in}(P, Q^T, \omega_{in}), \delta, \omega_{h_1}), \delta, \omega_{h_2}), \delta, \omega_{h_3}), \omega_{out}) \quad (15)$$

where, ω_x represents the model parameters of layer x of a multilayer perceptron. We used the MSE and the ReLU in our model [62]. The MSE determines the average squared difference between the target and predicted ratings, allowing

us to optimize our model's performance. On the other hand, the ReLU introduces non-linearity to our model. The first two models (M1 and M2) of the ensemble use one hidden layer and two hidden layers without dropout, and the third model, M3, uses three hidden layers and a dropout rate ($\delta=0.5$) to avoid overfitting [63]. In sub-model M3, certain nodes of the hidden layers are deactivated with the probability 'p' from the Bernoulli distribution. The output of each MLP is trained using a novel optimization technique called AdaMVRGO. The final predicted rating is the mean of the individual predictions of the models M1, M2, and M3. The objective function of each sub-model in the ensembled architecture is given in Eqs. (16), (17), and (18), respectively.

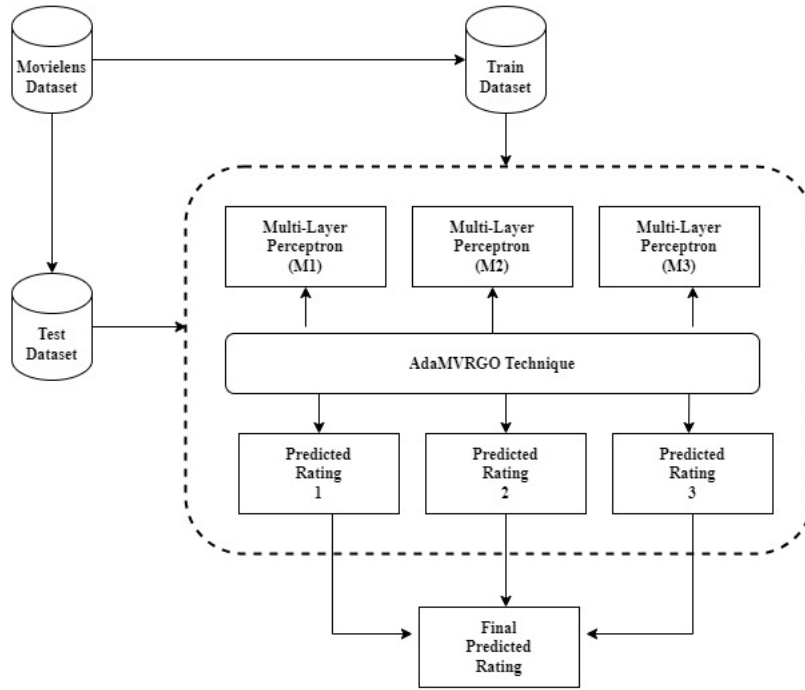


Figure 1. Deep CF framework

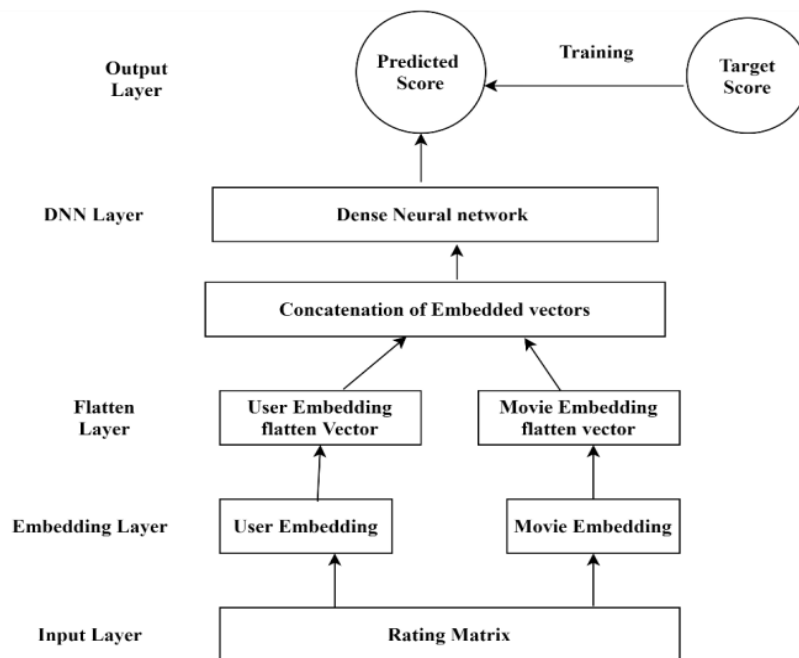


Figure 2. The basic structure of an MLP used in the ensemble

$$L_{M_1} = \min_{\omega_f} \sum_{(p_i, q_j) \in R} l(r_{ij}, \hat{r}_{ij}) + \gamma C(\omega_f) \quad (16)$$

$$L_{M_2} = \min_{\omega_f} \sum_{(p_i, q_j) \in R} l(r_{ij}, \hat{r}_{ij}) + \gamma C(\omega_f) \quad (17)$$

$$L_{M_3} = \min_{\omega_f} \sum_{(p_i, q_j) \in R} l(r_{ij}, \hat{r}_{ij}) + \gamma C_{dropout}(\omega_f) \quad (18)$$

where, $l(\cdot)$ represents the loss function, $\gamma > 0$ represents the step size, and $C(\omega_f)$ is the regularizer. The dropout is determined by using the Bernoulli distribution. The final rating is computed using Eq. (19).

$$\hat{r}_{ij} = average(L_{M_1} + L_{M_2} + L_{M_3}) \quad (19)$$

4.2 Learning using AdaMVRGO

Although ADAM is considered the best optimization algorithm for deep learning applications, it still suffers from slow convergence and generalization issues. Variance reduction methods are used to reduce the variance and achieve better generalization and a fast convergence rate. Variance reduction methods have recently become popular and are the best alternative to nonadaptive gradient methods such as SGD. These methods typically reduce the variance of stochastic gradients by taking a snapshot of the gradients for each 'm' optimization step. This snapshot's gradient information can be used to reduce the variance of subsequent smaller batch gradients [59]. SAG [64], SAGA [65], and SVRG [66] are the most standard variance reduction methods. The stochastic gradient variance is cancelled by SVRG with the control variate (the average of gradients computed at different snapshots), which has a zero exception. The proposed algorithm, AdaMVRGO, is developed using the ADAM and SVRG optimizers [11]. Algorithm 1 depicts the pseudocode for AdaMVRGO. Similar to SVRG, AdaMVRGO uses a nested looping construct. The outer loop consists of 'K' iterations, and the inner loop consists of 'm' iterations. The outermost loop determines the complete gradient at a random snapshot. This snapshot is updated after every 'm' step of parameter updating. In the inner loop, uniformly select a random data point and find the gradient g_t using the SVRG principle. This gradient is the variance-reduced gradient as shown in Eq. (20), which assists in approaching the optimal value with a constant step size δ [67]. The weight update rule is shown in Eq. (21).

$$g_t = \nabla L_{i_t}(x_t) - \nabla L_{i_t}(\omega_k) + \nabla L(\omega_k) \quad (20)$$

$$\omega_{t+1} = \omega_t - \delta g_t \quad (21)$$

Typically, the next snapshot point is set to the inner loop's final iteration value at the end of the inner loop [67], $\omega_{k+1} = \frac{1}{m} \sum_{t=1}^m x_t$. Adaptive optimization algorithms such as ADAM still suffer from convergence issues due to the presence of high variance. The proposed algorithm's change is that it reduces variance while calculating the gradient itself. The first and second moments are computed on these variance-reduced gradients.

The advantages of the AdaMVRGO algorithm compared to others are listed as follows:

- Experimentally, our proposed optimizer is robust compared to previous optimization approaches.

- Variance-reduced gradients are computed initially before smoothing them.
- The first and second moments are computed based on the variance-reduced gradients.
- The overall variance is reduced after some epochs, which results in fast convergence.

Algorithm 1: Pseudocode of AdaMVRGO algorithm

```

Initialize
 $\omega_0$  (starting point), K (outer loop), m (inner loop)
for s = 0 to K-1, do
Determine the total gradient of the objective function,  $\nabla L(\omega_s)$ 
Initialization: Initialize the first and second moment vectors
 $u_0 = 0$  and  $v_0 = 0$ , respectively, and the exponential decay rates for the first and second moments are  $\beta_1 = 0.9$  and  $\beta_2 = 0.999$ , respectively, decay = 0, learning rate  $\delta = 0.001$  and  $x^0 = \omega_s$ 
  for t = 1 to m do
    Randomly select a point  $i_t$ 
     $g_t = \nabla L_{i_t}(x_t) - \nabla L_{i_t}(\omega_s) + \nabla L(\omega_s)$ 
     $u_t = \beta_1 u_{t-1} + (1 - \beta_1) g_t$ 
     $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$ 
     $x_{t+1} = x_t - \delta g_t$ 
  end
   $\omega_{s+1} = \frac{1}{m} \sum_{t=1}^m x_t$ 
end
return  $\bar{\omega}_K = \frac{1}{K} \sum_{s=0}^{K-1} \omega_s$ 

```

The significance of the proposed framework over existing ones is given as follows:

- 1) In the proposed method, the ensemble of MLPs reduces the variance by taking the average of the prediction results of each MLP. This makes the model more generalized. It helps generate diversified recommendations.
- 2) In the novel optimizer, AdaMVRGO, variance-reduced gradients are used to compute the first and second moments. This helps in reducing the variance, which helps in fast convergence.

5. RESULTS AND DISCUSSION

The research concerns addressed in this section are [68]:

- **RQ1:** Does the proposed framework perform better than the baseline algorithms?
- **RQ2:** Does the proposed optimizer perform well compared to the other existing optimizers?

5.1 Data description

This paper evaluated the proposed framework using three standard datasets: M-100K, M-1M, and M-10M, taken from MovieLens's website [11]. The M-100K dataset has 100,000 ratings from 943 users and 1682 movies [69, 70]. The rating values range from 1 to 5. The information was gathered from 19th September 1997 to 22nd April 1998. This dataset contains basic demographic information about the user, like age, gender, occupation, etc. Every user in this collection has reviewed a minimum of 20 movies, and the users who had fewer than 20 ratings are not considered in the dataset. Also, users whose demographic information is not present are

excluded from this collection. The movies present in the dataset belong to any of the 19 genres: unknown, action, adventure, animation, children's, comedy, crime, documentary, drama, fantasy, film noir, horror, musical, mystery, romance, sci-fi, thriller, war, and western.

The M-1M dataset has 1,000,209 ratings for 3900 movies submitted by 6040 users collected during the year 2000 [11, 71]. Every user has reviewed at least 20 movies. The dataset consists of demographic information about the user, like gender, age, and occupation. The movies in the dataset fall under different genres like action, adventure, animation, children's, comedy, etc. [72].

The M-10M dataset has 1,00,00,054 ratings for 10,681 movies submitted by 71,567 users [11]. This dataset uses a 5-star scale with an increment of half a star. Unlike the previous datasets, no demographic information about the user is provided. The movies in the dataset fall under different genres like action, adventure, animation, children's, comedy, etc. [72]. Each user rates at least 20 movies [56]. Table 1 shows the statistical analysis of all three datasets.

Table 1. MovieLens dataset statistics [73]

	ML - 100K [73]	ML - 1M [73]	ML - 10M [73]
# Users	943	6040	71,567
#Movies	1682	3900	10,681
Total # Ratings	1,00,000	10,00,209	1,00,00,054
Sparsity	93.70%	93.53%	98.66%
Mean	3.54	3.58	3.52
Standard Deviation	1.06	1.11	1.05
Minimum Rating	0.5	1	0.5
Maximum Rating	5	5	5
25% of Ratings	3	3	3
50% of Ratings	4	4	4
75% of Ratings	4	4	4

5.2 Experimental setup

The following system requirements are used to implement the proposed framework: Language: Python 3.7, OS: Windows 11 (64-bit), RAM: 16 GB, graphics card: NVIDIA GeForce GTX 1650, and processor: AMD Ryzen 5 4600H with Radeon Graphics [72, 74]. The proposed framework is implemented based on TensorFlow 1.14 and the Keras 2.3.1 framework. Experiments were carried out with 10-fold cross-validation [75]. The following hyperparameters are used during experimentation: number of latent factors: [8, 16, 32, 64], batch size: [64, 128, 256], number of epochs: [50, 100], learning rate: 0.001, decay rate: 0.01, dropout: 0.5.

5.3 Evaluation metrics

The metrics used to evaluate the proposed model are MSE, RMSE, and MAE. MSE is the mean squared error obtained from the difference between the predicted and actual ratings, and RMSE is the square root of MSE [11]. The formulae for MSE and RMSE are given in Eqs. (22) and (23).

$$MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2 \quad (22)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2} \quad (23)$$

MAE is the mean absolute difference between the predicted and actual ratings. The formula for MAE is shown in Eq. (24).

$$MAE = \frac{1}{n} \sum_{i=1}^n |Y_i - \hat{Y}_i| \quad (24)$$

where, \hat{Y}_i is the predicted rating and Y_i is the actual rating, n represent some observations. Smaller values of Eqs. (22), (23), and (24) mean higher accuracy.

5.4 Experiments and discussion

5.4.1 Comparative analysis with the existing techniques (RQ1)

Using the M-1M and M-10M datasets, this section compares the proposed framework Deep CF-AdaMVRGO to six existing models [24-29] in terms of RMSE. The research findings proved that the proposed model performed better than the existing models with different hyperparameters [76], like epochs and batch size. To prove the proposed model is statistically significant, a paired t-test with a 0.05 level of significance is used, as shown in Section 5.4.3. The quantitative comparison results are depicted in Tables 2 and 3. The existing models for comparison are:

- Multiviews NN [25] initially learned the low-dimensional user and item vectors separately. A feed-forward neural network (FFNN) is used to learn the nonlinear interactions between users and items for rating prediction [77].
- NCF [26] implemented a neural network-based matrix factorization model [78], and the loss was adjusted to a squared loss for rating prediction.
- SemRe-DCF [27] combines a rating matrix with movie plot text, and a denoising autoencoder is used for predicting the ratings.
- DPGMF [24] fused plot texts into ratings and proposed a deep-plot-aware generalised matrix factorization to predict missing ratings.
- DELCR [28] uses a DNN to train the user and item embeddings separately to make a latent factor model for collaborative filtering that is based on deep learning.
- DLFCF [29] uses a two-level deep learning model to learn about the rating matrix's hidden features.

From Tables 2 and 3, it is evident that the proposed model has given a considerably reduced RMSE when compared with the existing methods using M-1M and M-10M datasets. While experimenting, it was also noted that the proposed algorithm showed better results on the M-1M dataset with 100 epochs and 128 batch size. Similarly, it showed better results on the M-10M dataset with 50 epochs and 256 batch size.

The significant benefit of the proposed model over the existing models [24-29] is that it allows the model to make accurate predictions even for new movies that have limited or no information available. Additionally, the proposed framework outperforms the existing models in terms of reducing biases and variances in the ensemble, which results in more reliable and accurate predictions.

5.4.2 Comparative analysis with the existing optimizers (RQ2)

This section evaluates the performance of the proposed

optimizer AdaMVRGO on the ensemble framework with current optimizers such as Adagrad, RMSProp, ADAM, and

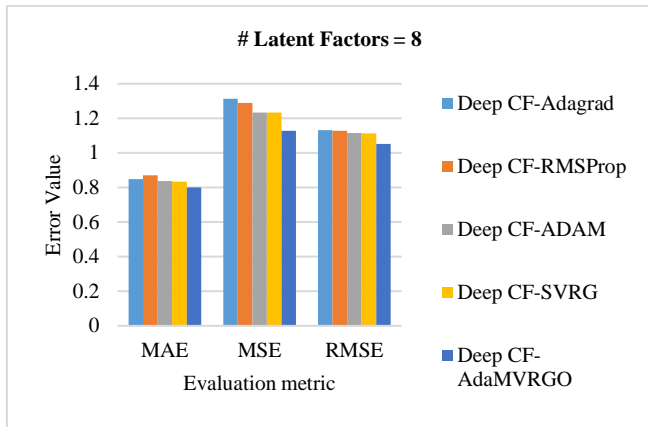
SVRG on M-100K, M-1M, and M-10M datasets in terms of RMSE, MAE, and MSE metrics [11].

Table 2. Quantitative comparison results in terms of RMSE using the M-1M dataset (A lower RMSE value is better.)

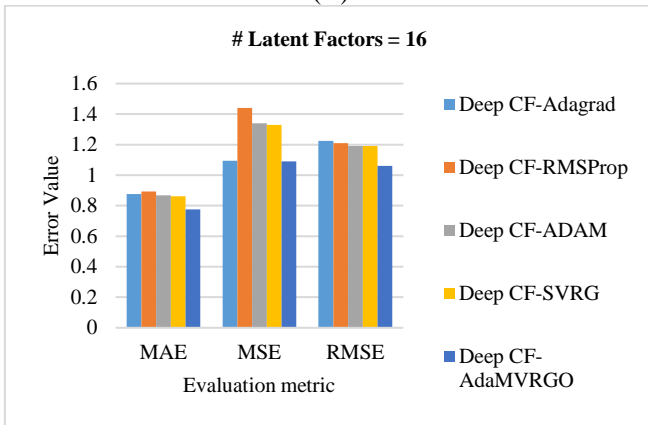
Parameters	Multiviews NN [25]	NCF [26]	SemRe-DCF [27]	DPGMF [24]	DELRCR [28]	DLFCF [29]	Proposed Model
#Epochs 50	0.857	0.971	0.867	0.861	0.884	0.877	0.852
Batch Size 64							
% decrease in RMSE when compared with the proposed model	0.583	12.255	1.730	1.045	3.619	2.850	
#Epochs 50	0.841	0.967	0.858	0.849	0.879	0.876	0.837
Batch Size 128							
% decrease in RMSE when compared with the proposed model	0.475	13.444	2.447	1.413	4.778	4.452	
#Epochs 50	0.849	0.946	0.853	0.857	0.854	0.871	0.833
Batch Size 256							
% decrease in RMSE when compared with the proposed model	1.884	11.945	2.344	2.8	2.459	4.362	
#Epochs 100	0.834	0.942	0.851	0.849	0.842	0.868	0.829
Batch Size 64							
% decrease in RMSE when compared with the proposed model	0.599	11.996	2.585	2.355	1.543	4.493	
#Epochs 100	0.833	0.938	0.845	0.821	0.818	0.866	0.816
Batch Size 128							
% decrease in RMSE when compared with the proposed model	2.040	13.006	3.432	0.609	0.244	5.773	
#Epochs 100	0.831	0.941	0.844	0.834	0.837	0.875	0.824
Batch Size 256							
% decrease in RMSE when compared with the proposed model	1.787	12.434	2.369	1.199	1.553	5.828	

Table 3. Quantitative comparison results in terms of RMSE using the M-10M dataset (A lower RMSE value is better.)

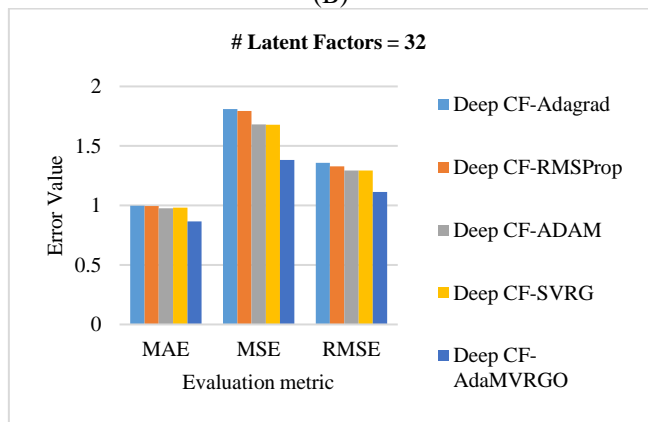
Parameters	Multiviews NN [25]	NCF [26]	SemRe-DCF [27]	DPGMF [24]	DELRCR [28]	DLFCF [29]	Proposed Model
#Epochs 50	0.819	0.912	0.832	0.783	0.787	0.799	0.781
Batch Size 64							
% decrease in RMSE when compared with the proposed model	4.639	14.364	6.129	0.255	0.762	2.252	
#Epochs 50	0.779	0.903	0.812	0.798	0.779	0.804	0.777
Batch Size 128							
% decrease in RMSE when compared with the proposed model	0.256	13.953	4.310	2.631	0.256	3.358	
#Epochs 50	0.776	0.899	0.773	0.765	0.774	0.798	0.761
Batch Size 256							
% decrease in RMSE when compared with the proposed model	1.933	15.35	1.552	0.522	1.679	4.636	
#Epochs 100	0.777	0.910	0.798	0.768	0.777	0.819	0.765
Batch Size 64							
% decrease in RMSE when compared with the proposed model	1.544	15.934	4.135	0.390	1.544	6.593	
#Epochs 100	0.778	0.921	0.829	0.769	0.821	0.818	0.768
Batch Size 128							
% decrease in RMSE when compared with the proposed model	1.285	16.612	7.358	0.130	6.455	6.112	
#Epochs 100	0.779	0.928	0.867	0.771	0.837	0.820	0.770
Batch Size 256							
% decrease in RMSE when compared with the proposed model	1.155	17.026	11.188	0.129	8.004	6.097	



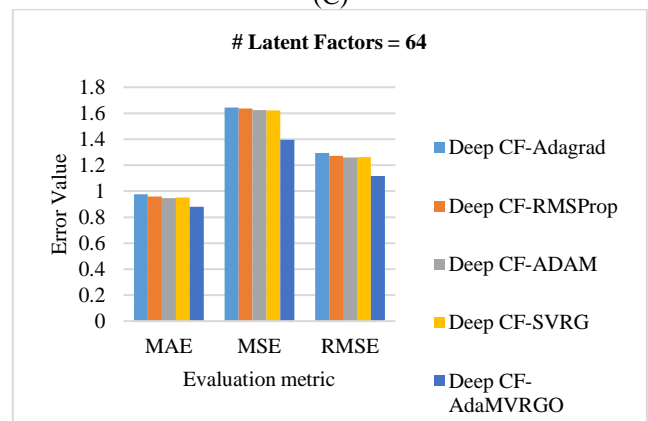
(A)



(B)



(C)



(D)

Figure 3. Performance of the Deep CF-AdaMVRGO algorithm using M-100K (A to D from top to bottom)

Analysis of the M-100K dataset: Using 10-fold cross-validation on the M-100K dataset, the proposed Deep CF-AdaMVRGO method is compared with other methods like Deep CF-Adagrad, Deep CF-RMSProp, Deep CF-Adam, and Deep CF-SVRG in terms of RMSE, MSE, and MAE metrics [78, 79]. Experiments were carried out with various settings [78, 79]. Experiments were carried out with various settings, such as a learning rate of 0.001, a batch size of 256, 100 epochs, and latent factors [8, 16, 32, 64]. Figure 3 depicts the performance of the Deep CF-AdaMVRGO method over the other methods on the M-100K dataset with MAE, MSE, and RMSE for different latent factors. [3(A) compares the evaluation metric values of different optimizers when the latent factors are 8; 3(B) compares the evaluation metric values of different optimizers when the latent factors are 16; 3(C) compares the evaluation metric values of different optimizers when the latent factors are 32; 3(D) compares the evaluation metric values of different optimizers when the latent factors are 64].

The total number of computations is represented in floating point operations (FLOPs), and it is observed that the proposed algorithm took fewer FLOPs to reach the optimum when compared with other algorithms, as shown graphically in Figure 4.

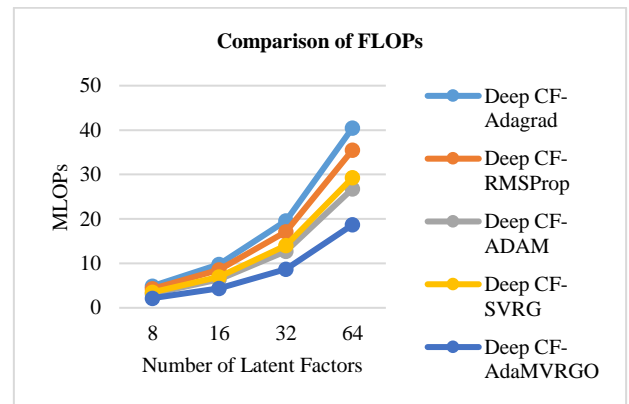
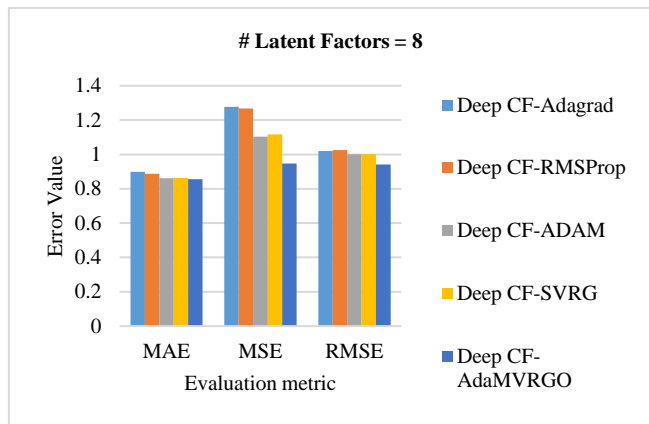
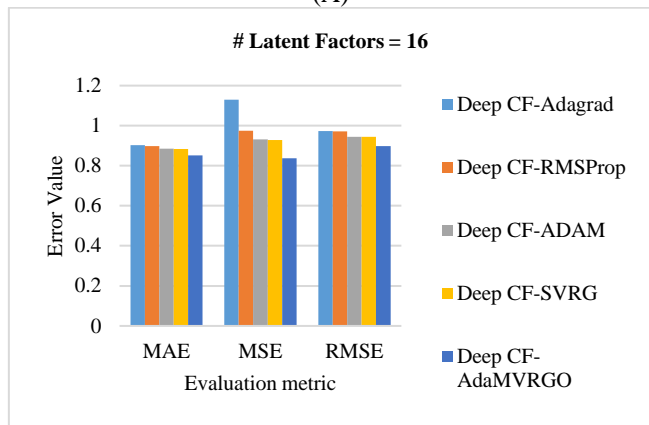


Figure 4. Comparison of MFLOPs using the M-100K dataset

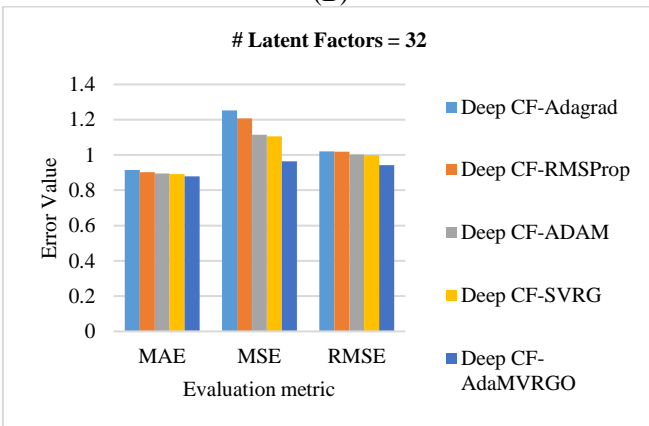
Analysis of the M-1M dataset: Using 10-fold cross-validation on the M-1M dataset, the proposed Deep CF-AdaMVRGO method is compared with other methods like Deep CF-Adagrad, Deep CF-RMSProp, Deep CF-Adam, and Deep CF-SVRG in terms of RMSE, MSE, and MAE metrics [78, 79]. Experiments were carried out with various settings, such as a learning rate of 0.001, a batch size of 256, 100 epochs, and latent factors [8, 16, 32, 64]. Figure 5 depicts the performance of the Deep CF-AdaMVRGO method over the other methods on the M-1M dataset with MAE, MSE, and RMSE for different latent factors. [5(A) compares the evaluation metric values of different optimizers when the latent factors are 8, 5(B) compares the evaluation metric values of different optimizers when the latent factors are 16, 5(C) compares the evaluation metric values of different optimizers when the latent factors are 32, 5(D) compares the evaluation metric values of different optimizers when the latent factors are 64]. The total number of computations is represented in FLOPs, and it is observed that the proposed algorithm took fewer FLOPs to reach the optimum when compared with other algorithms, as shown graphically in Figure 6.



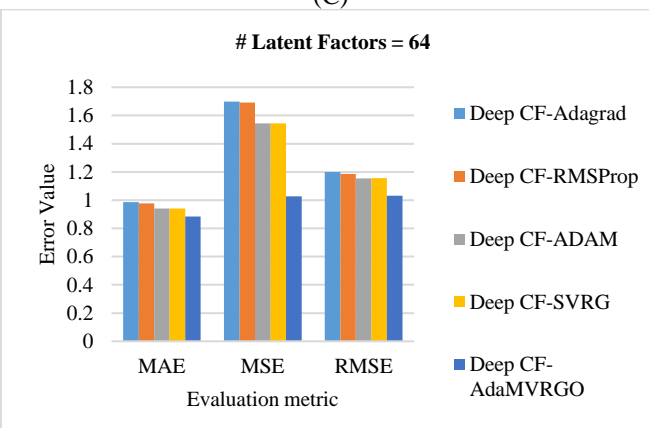
(A)



(B)



(C)



(D)

Figure 5. Performance of the Deep CF-AdaMVRGO algorithm using M-1M dataset (A to D from top to bottom)

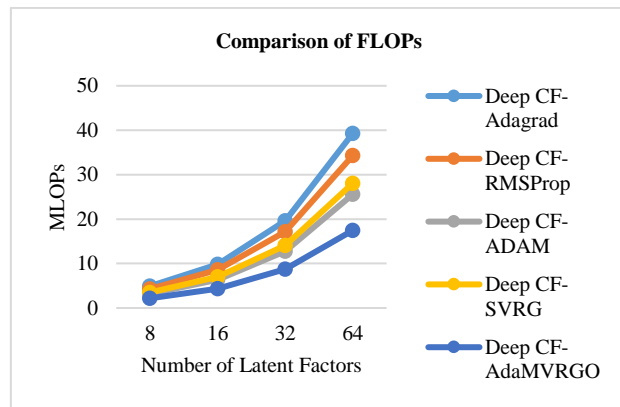
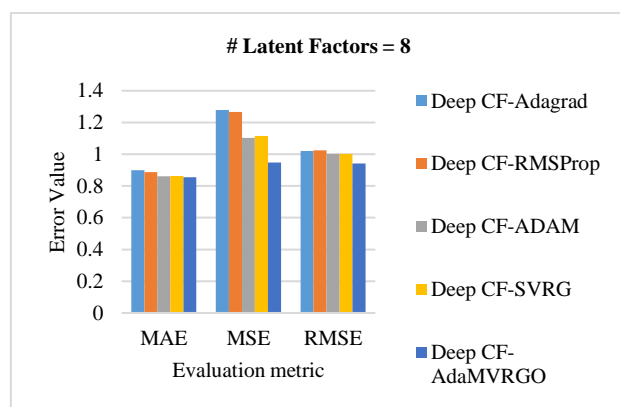
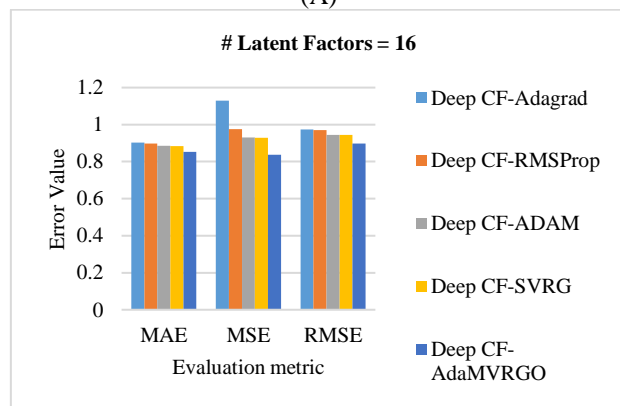


Figure 6. Comparison of MFLOPs using the M-1M dataset

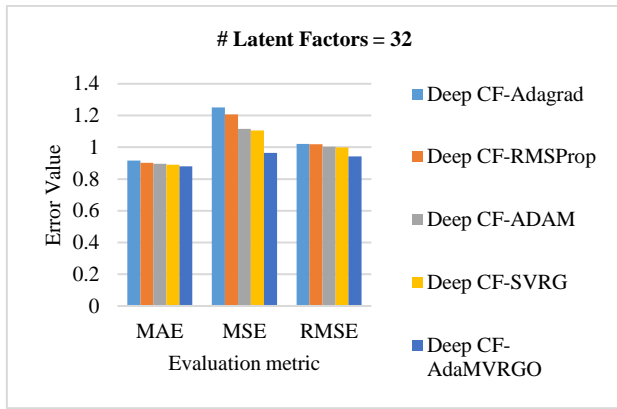
Analysis of the M-10M dataset: Using 10-fold cross-validation on the M-10M dataset, the proposed Deep CF-AdaMVRGO method is compared with other methods like Deep CF-Adagrad, Deep CF-RMSProp, Deep CF-Adam, and Deep CF-SVRG in terms of RMSE, MSE, and MAE metrics [78, 79]. Experiments were carried out with various settings, such as a learning rate of 0.001, a batch size of 256, 100 epochs, and latent factors [8, 16, 32, 64]. Figure 7 depicts the performance of the Deep CF-AdaMVRGO method over the other methods on the ML 10M dataset with MAE, MSE, and RMSE for different latent factors. [7(A) compares the evaluation metric values of different optimizers when the latent factors are 8, 7(B) compares the evaluation metric values of different optimizers when the latent factors are 16, 7(C) compares the evaluation metric values of different optimizers when the latent factors are 32, and 7(D) compares the evaluation metric values of different optimizers when the latent factors are 64].



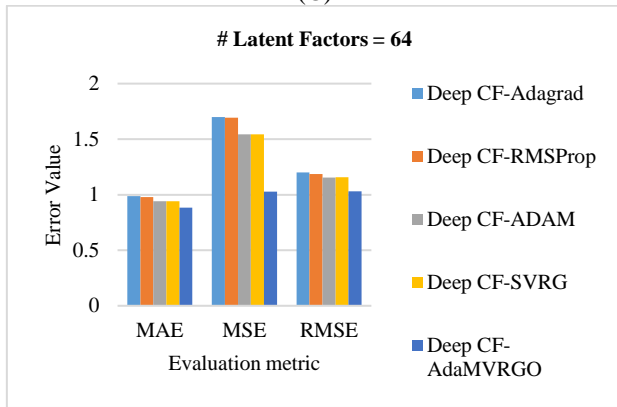
(A)



(B)



(C)



(D)

Figure 7. Performance of the deep CF-AdaMVRGO algorithm using M-10M dataset (A to D from top to bottom)

The total number of computations is represented in FLOPs, and it is observed that the proposed algorithm took fewer FLOPs to reach the optimum when compared with other algorithms, as shown graphically in Figure 8.

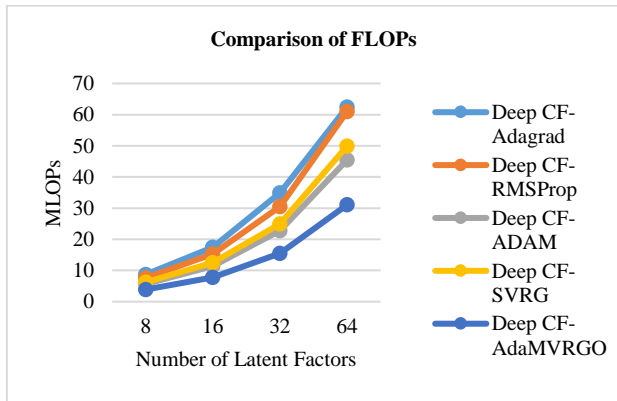


Figure 8. Comparison of MFLOPs using the M-10M dataset

From Figures 3, 5, and 7, it is proven that the proposed novel optimizer, AdaMVRGO, outperformed the existing optimisation techniques on the ensemble framework in terms of all evaluation metrics [80]. From Figures 4, 6, and 8, it can be observed that AdaMVRGO converged significantly faster than the existing optimization techniques. This suggests that AdaMVRGO not only produces superior results in terms of accuracy metrics but also does so in a more efficient manner. These findings highlight the effectiveness of AdaMVRGO in improving the performance of the ensemble framework and make it a promising choice for future applications.

5.4.3 Statistical significance and discussions

To verify the results of the proposed model over the existing models, statistical analysis was performed on the experimental results using a paired t-test for the M-1M and M-10M datasets. The statistical analysis is performed at a 0.05 (95%) level of significance, and the results are analysed in terms of a t-value. The proposed model is considered robust when the p-value is less than the significance level [81]. From Table 4, we can conclude that the proposed model is significantly stronger with a low p-value compared to the existing models at 9 degrees of freedom. These findings suggest that the proposed model is highly reliable and provides a significant improvement over the existing models in predicting the observed data. Moreover, the low p-value indicates that the results are unlikely to occur by chance alone, further validating the proposed models' robustness [75]. Overall, these findings offer compelling proof of the proposed model's efficacy and demonstrate that it outperforms existing models in terms of statistical strength and accuracy [82].

Table 4. Statistical significance on M-1M & M-10M datasets [83]

	M-1M	M-10M
MultiviewsNN [25]	0.015924019	0.011271812
NCF [26]	0.000003159	0.000014055
SemRe-DCF [27]	0.000187744	0.031027946
DPGMF [24]	0.014531813	0.189003658
DELCR [28]	0.052601661	0.083751047
DLFCF [29]	0.00001175	0.000986522
Proposed Model	0.000001579	0.000007028

6. CONCLUSION

Deep collaborative filtering has proven to be one of the best recommendation techniques over the other latent factor models to capture hidden, nonlinear, and complex relationships between users and movies. To handle the sparsity issue and enhance movie recommendation accuracy, we developed an ensemble framework called Deep CF-AdaMVRGO, which uses three sub-models: M1, M2, and M3. The first two models (M1 and M2) of the ensemble use one hidden layer and two hidden layers without dropout, and the third model, M3, uses three hidden layers and a dropout rate taken from the Bernoulli distribution to avoid overfitting. The output of each MLP is trained using a novel optimization technique called AdaMVRGO. Taking the average of the individual predictions made by the models M1, M2, and M3 yields the final rating.

Experiments were conducted using the M-100K, M-1M, and M-10M datasets in terms of evaluation metrics like MSE, RMSE, and MAE. Simulation results showed that the new optimizer AdaMVRGO had a low error value in recommending movies, compared to optimizers like Adagrad, RMSProp, ADAM, and SVRG on the proposed ensemble architecture. Also, the proposed optimizer achieved the optimum with fewer FLOPs. The experimental results showed that the Deep CF-AdaMVRGO performed well over the benchmark algorithms on both the M-1M and M-10M datasets in terms of the RMSE value. Experiments were done with different parameters, and the results showed that the proposed model is better than the existing models on the M-1M dataset when the number of epochs was 100 and the batch size was 128. Also, the proposed model showed a 2.040%, 13.006%,

3.432%, 0.609%, 0.244%, and 5.773% decrease in terms of RMSE value over the baseline models. In the same way, it did better on the M-10M dataset when the number of epochs was 50 and the batch size was 256. The proposed algorithm did better than the baseline models in terms of RMSE value by 1.933%, 15.35%, 1.552%, 0.522%, 1.679%, and 4.636%. From the simulations, we infer that the ensemble-based model can generate more accurate recommendations than the individual methods.

In this paper, the final prediction is the mean of the predictions obtained from the three MLPs, which may not be the best all the time. Future enhancements may include the use of stacking ensembles to construct models with low bias and variance; the addition of side information to the movies like their genres, user reviews, implicit feedback, etc. to enhance the recommendation accuracy; and the use of an attribute selection algorithm to improve movie recommendation performance even further.

DECLARATION

CRedit authorship contribution statement: V. Lakshmi Chetana: Conceptualization, Methodology, and Writing—Original Draft

Hari Seetha: supervision, writing (review and editing).

Funding: This research received no external funding. This work is carried out as part of my doctoral committee.

Conflict of Interest: The authors declare no conflicts of interest.

Data Availability: <https://groupLens.org/datasets/movielens/>.

REFERENCES

- [1] Deldjoo, Y., Dacrema, M.F., Constantin, M.G., Eghbal-Zadeh, H., Cereda, S., Schedl, M., Ionescu, B., Cremonesi, P. (2019). Movie genome: Alleviating new item cold start in movie recommendation. *User Modeling and User-Adapted Interaction*, 29: 291-343. <https://doi.org/10.1007/s11257-019-09221-y>
- [2] Chen, G., Jing, W., Wen, X., Lu, Z., Zhao, S. (2021). An edge caching strategy based on separated learning of user preference and content popularity. In *2021 IEEE/CIC International Conference on Communications in China (ICCC)*, Xiamen, China, pp. 1018-1023. <https://doi.org/10.1109/ICCC52777.2021.9580288>
- [3] Shahbazi, Z., Hazra, D., Park, S., Byun, Y.C. (2020). Toward improving the prediction accuracy of product recommendation system using extreme gradient boosting and encoding approaches. *Symmetry*, 12(9): 1566. <https://doi.org/10.3390/sym12091566>
- [4] Tahmasebi, H., Ravanmehr, R., Mohamadrezai, R. (2021). Social movie recommender system based on deep autoencoder network using Twitter data. *Neural Computing and Applications*, 33: 1607-1623. <https://doi.org/10.1007/s00521-020-05085-1>
- [5] Qu, W., Song, K.S., Zhang, Y.F., Feng, S., Wang, D.L., Yu, G. (2013). A novel approach based on multi-view content analysis and semi-supervised enrichment for movie recommendation. *Journal of Computer Science and Technology*, 28(5): 776-787. <https://doi.org/10.1007/s11390-013-1376-7>
- [6] Ricci, F., Rokach, L., Shapira, B. (2015). *Recommender Systems Handbook*. Springer.
- [7] Widiyaningtyas, T., Hidayah, I., Adji, T.B. (2021). User profile correlation-based similarity (UPCSim) algorithm in movie recommendation system. *Journal of Big Data*, 8: 1-21. <https://doi.org/10.1186/s40537-021-00425-x>
- [8] Ricci, F., Rokach, L., Shapira, B. (Eds.). (2022). *Recommender Systems Handbook*. Springer New York, NY. <https://doi.org/10.1007/978-1-0716-2197-4>
- [9] Al-Shamri, M.Y.H., Bharadwaj, K.K. (2008). Fuzzy-genetic approach to recommender systems based on a novel hybrid user model. *Expert systems with applications*, 35(3): 1386-1399. <https://doi.org/10.1016/j.eswa.2007.08.016>
- [10] Christou, I.T., Amolochitis, E., Tan, Z.H. (2016). AMORE: Design and implementation of a commercial-strength parallel hybrid movie recommendation engine. *Knowledge and Information Systems*, 47: 671-696. <https://doi.org/10.1007/s10115-015-0866-z>
- [11] Lakshmi Chetana, V., Seetha, H. (2022). CF-AMVRGO: Collaborative filtering based adaptive moment variance reduction gradient optimizer for movie recommendations. *International Journal of Computers and Applications*, 44(11): 1015-1023. <https://doi.org/10.1080/1206212X.2022.2097769>
- [12] Chetana, V.L., Batchu, R.K., Devarasetty, P., Voddelli, S., Dalli, V.P. (2023). Effective movie recommendation based on improved densenet model. *Multiagent and Grid Systems*, 19(2): 133-147. <https://doi.org/10.3233/mgs-230012>
- [13] Vellino, A., Zeber, D. (2007). A hybrid, multi-dimensional recommender for journal articles in a scientific digital library. In *2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Workshops*, Silicon Valley, CA, USA, pp. 111-114. <https://doi.org/10.1109/WI-IATW.2007.29>
- [14] Weng, L.T., Xu, Y., Li, Y., Nayak, R. (2005). An improvement to collaborative filtering for recommender systems. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, Vienna, pp. 792-795. <https://doi.org/10.1109/CIMCA.2005.1631361>
- [15] Rawat, R.M., Tomar, V., Kumar, V. (2020). An embedding-based deep learning approach for movie recommendation. In *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, Coimbatore, India, pp. 1145-1150. <https://doi.org/10.1109/ICCES48766.2020.9137998>
- [16] Choi, S.M., Ko, S.K., Han, Y.S. (2012). A movie recommendation algorithm based on genre correlations. *Expert Systems with Applications*, 39(9): 8079-8085. <https://doi.org/10.1016/j.eswa.2012.01.132>
- [17] Bi, Y., Williams, M.A. (2010). *Knowledge Science, Engineering and Management. Lecture Notes in Computer Science*, 6291. <https://doi.org/10.1007/978-3-642-15280-1>
- [18] Zhang, Y., Zhang, M., Liu, Y., Ma, S., Feng, S. (2013). Localized matrix factorization for recommendation based on matrix block diagonal forms. In *Proceedings of the 22nd International Conference on World Wide Web*, pp. 1511-1520.

- <https://doi.org/10.1145/2488388.2488520>
- [19] Li, L., Huang, H., Li, Q., Man, J. (2023). Personalized movie recommendations based on deep representation learning. *PeerJ Computer Science*, 9: e1448. <https://doi.org/10.7717/peerj-cs.1448>
- [20] Motahhir, S., Bossoufi, B. (2023). *Digital Technologies and Applications*. SpringerLink. <https://doi.org/10.1007/978-3-031-29860-8>
- [21] Fu, Z., Niu, X., Maher, M.L. (2023). Deep learning models for serendipity recommendations: A survey and new perspectives. *ACM Computing Surveys*, 56(1): 1-26. <https://doi.org/10.1145/3605145>
- [22] What Is an AI-Generated Artwork? https://pure.rug.nl/ws/files/622700064/ROBO_AI_Con2023_Abstract_Book.pdf.
- [23] Liu, M., Zhang, W., Orabona, F., Yang, T. (2020). Adam⁺: A stochastic method with adaptive variance reduction. *arXiv preprint arXiv:2011.11985*. <https://doi.org/10.48550/arXiv.2011.11985>
- [24] Sun, X., Zhang, H., Wang, M., Yu, M., Yin, M., Zhang, B. (2020). Deep plot-aware generalized matrix factorization for collaborative filtering. *Neural Processing Letters*, 52: 1983-1995. <https://doi.org/10.1007/s11063-020-10333-5>
- [25] Fu, M., Qu, H., Yi, Z., Lu, L., Liu, Y. (2018). A novel deep learning-based collaborative filtering model for recommendation system. *IEEE Transactions on Cybernetics*, 49(3): 1084-1096. <https://doi.org/10.1109/TCYB.2018.2795041>
- [26] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S. (2017). Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, pp. 173-182. <https://doi.org/10.1145/3038912.3052569>
- [27] Yue, L., Sun, X.X., Gao, W.Z., Feng, G.Z., Zhang, B.Z. (2018). Multiple auxiliary information based deep model for collaborative filtering. *Journal of Computer Science and Technology*, 33(1): 668-681. <https://doi.org/10.1007/s11390-018-1848-x>
- [28] Tegene, A., Liu, Q., Gan, Y., Dai, T., Leka, H., Ayenew, M. (2023). Deep learning and embedding based latent factor model for collaborative recommender systems. *Applied Sciences*, 13(2): 726. <https://doi.org/10.3390/app13020726>
- [29] Mongia, A., Jhamb, N., Chouzenoux, E., Majumdar, A. (2020). Deep latent factor model for collaborative filtering. *Signal Processing*, 169: 107366. <https://doi.org/10.1016/j.sigpro.2019.107366>
- [30] Fu, M., Agrawal, A., Irissappane, A.A., Zhang, J., Huang, L., Qu, H. (2021). Deep reinforcement learning framework for category-based item recommendation. *IEEE Transactions on Cybernetics*, 52(11): 12028-12041. <https://doi.org/10.1109/TCYB.2021.3089941>
- [31] Huang, T., Zhang, D., Bi, L. (2020). Neural embedding collaborative filtering for recommender systems. *Neural Computing and Applications*, 32: 17043-17057. <https://doi.org/10.1007/s00521-020-04920-9>
- [32] Feng, C., Liang, J., Song, P., Wang, Z. (2020). A fusion collaborative filtering method for sparse data in recommender systems. *Information Sciences*, 521: 365-379. <https://doi.org/10.1016/j.ins.2020.02.052>
- [33] Xue, F., He, X., Wang, X., Xu, J., Liu, K., Hong, R. (2018). Deep item-based collaborative filtering for Top-N recommendation. *arXiv preprint arXiv:1811.04392*. <https://doi.org/10.48550/arXiv.1811.04392>
- [34] Sun, X., Zhang, L., Wang, Y., Yu, M., Yin, M., Zhang, B. (2021). Attribute-aware deep attentive recommendation. *The Journal of Supercomputing*, 77: 5510-5527. <https://doi.org/10.1007/s11227-020-03459-9>
- [35] Mu, R., Zeng, X. (2018). Collaborative filtering recommendation algorithm based on knowledge graph. *Mathematical Problems in Engineering*, 2018: 9617410. <https://doi.org/10.1155/2018/9617410>
- [36] Ji, K., Shen, H. (2015). Making recommendations from top-N user-item subgroups. *Neurocomputing*, 165: 228-237. <https://doi.org/10.1016/j.neucom.2015.03.013>
- [37] Ji, K., Yuan, Y., Ma, K., Sun, R., Chen, Z., Wu, J. (2019). Context-aware recommendations via a tree-based ensemble framework. In *Proceedings of the ACM Turing Celebration Conference-China*, pp. 1-5. <https://doi.org/10.1145/3321408.3322839>
- [38] Neera, J., Chen, X., Aslam, N., Wang, K., Shu, Z. (2021). Private and utility enhanced recommendations with local differential privacy and gaussian mixture model. *IEEE Transactions on Knowledge and Data Engineering*, 35(4): 4151-4163. <https://doi.org/10.1109/TKDE.2021.3126577>
- [39] Berlioz, A., Friedman, A., Kaafar, M.A., Boreli, R., Berkovsky, S. (2015). Applying differential privacy to matrix factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pp. 107-114. <https://doi.org/10.1145/2792838.2800173>
- [40] Duma, M., Twala, B. (2019). Sparseness reduction in collaborative filtering using a nearest neighbour artificial immune system with genetic algorithms. *Expert Systems with Applications*, 132: 110-125. <https://doi.org/10.1016/j.eswa.2019.04.034>
- [41] Huang, H., Wei, Y., Yuan, X., Zheng, R. (2023). Weighted matrix factorization with wilson lower bound score. In *Proceedings of the 2023 5th Asia Pacific Information Technology Conference*, pp. 33-37. <https://doi.org/10.1145/3588155.3588160>
- [42] Shen, X.J., Ni, C., Wang, L., Zha, Z.J. (2021). Sliker: Sparse loss induced kernel ensemble regression. *Pattern Recognition*, 109: 107587. <https://doi.org/10.1016/j.patcog.2020.107587>
- [43] Yang, L., Shami, A., Stevens, G., De Rusett, S. (2022). LCCDE: A decision-based ensemble framework for intrusion detection in the internet of vehicles. In *GLOBECOM 2022-2022 IEEE Global Communications Conference*, Rio de Janeiro, Brazil, pp. 3545-3550. <https://doi.org/10.1109/GLOBECOM48099.2022.10001280>
- [44] Sharma, S., Gupta, V., Mudgal, D., Srivastava, V. (2023). Predicting biomechanical properties of additively manufactured polydopamine coated poly lactic acid bone plates using deep learning. *Engineering Applications of Artificial Intelligence*, 124: 106587. <https://doi.org/10.1016/j.engappai.2023.106587>
- [45] Blockeel, H., Kersting, K., Nijssen, S., Železný, F. (Eds). (2013). *Machine learning and knowledge discovery in databases*. SpringerLink. <https://doi.org/10.1007/978-3-642-40988-2>
- [46] Monish, H., Pandey, A.C. (2020). A comparative assessment of data mining algorithms to predict fraudulent firms. In *2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, Noida, India, pp. 117-122. <https://doi.org/10.1109/Confluence47617.2020.9057968>

- [47] Li, D., Chen, C., Lv, Q., Gu, H., Lu, T., Shang, L., Gu, N., Chu, S.M. (2018). AdaError: An adaptive learning rate method for matrix approximation-based collaborative filtering. In Proceedings of the 2018 World Wide Web Conference, pp. 741-751. <https://doi.org/10.1145/3178876.3186155>
- [48] Min, E., Long, J., Cui, J. (2018). Analysis of the variance reduction in SVRG and a new acceleration method. *IEEE Access*, 6: 16165-16175. <https://doi.org/10.1109/ACCESS.2018.2814212>
- [49] Ming, Y., Zhao, Y., Wu, C., Li, K., Yin, J. (2018). Distributed and asynchronous stochastic gradient descent with variance reduction. *Neurocomputing*, 281: 27-36. <https://doi.org/10.1016/j.neucom.2017.11.044>
- [50] Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In: Lechevallier, Y., Saporta, G. (eds) Proceedings of COMPSTAT'2010. Physica-Verlag HD. https://doi.org/10.1007/978-3-7908-2604-3_16
- [51] Kingma, D.P., Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. <https://doi.org/10.48550/arXiv.1412.6980>
- [52] Stephanidis, C., Antona, M., Ntoa, S. (Eds.). (2021). HCI International 2021 – Posters. SpringerLink. <https://doi.org/10.1007/978-3-030-78642-7>
- [53] Duchi, J., Hazan, E., Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7): 2121-2159.
- [54] Tieleman, T., Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 4(2): 26-31.
- [55] Muneer, A., Taib, S.M., Fati, S.M., Balogun, A.O., Aziz, I.A. (2022). A hybrid deep learning-based unsupervised anomaly detection in high dimensional data. *Computers, Materials & Continua*, 70(3): 5363-5381. <https://doi.org/10.32604/cmc.2022.021113>
- [56] Dubey, S.R., Chakraborty, S., Roy, S.K., Mukherjee, S., Singh, S.K., Chaudhuri, B.B. (2019). diffGrad: An optimization method for convolutional neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 31(11): 4500-4511. <https://doi.org/10.1109/TNNLS.2019.2955777>
- [57] Wilson, A.C., Roelofs, R., Stern, M., Srebro, N., Recht, B. (2017). The marginal value of adaptive gradient methods in machine learning. *Advances in Neural Information Processing Systems*, 30.
- [58] McVinish, R., Mengersen, K., Nur, D., Rousseau, J., Guihenneuc-Jouyaux, C. (2013). Recentered importance sampling with applications to Bayesian model validation. *Journal of Computational and Graphical Statistics*, 22(1): 215-228. <https://doi.org/10.1080/10618600.2012.681239>
- [59] Elibol, M., Lei, L., Jordan, M. I. (2020). Variance reduction with sparse gradients. arXiv preprint arXiv:2001.09623. <https://doi.org/10.48550/arXiv.2001.09623>
- [60] Li, J., Xu, W., Wan, W., Sun, J. (2018). Movie recommendation based on bridging movie feature and user interest. *Journal of Computational Science*, 26: 128-134. <https://doi.org/10.1016/j.jocs.2018.03.009>
- [61] Wu, L., Quan, C., Li, C., Wang, Q., Zheng, B., Luo, X. (2019). A context-aware user-item representation learning for item recommendation. *ACM Transactions on Information Systems (TOIS)*, 37(2): 1-29. <https://doi.org/10.1145/3298988>
- [62] Salha-Galvan, Guillaume. (2022). Contributions to Representation Learning with Graph Autoencoders and Applications to Music Recommendation.
- [63] Khanna, A., Gupta, D., Bhattacharyya, S., Snasel, V., Platos, J., Hassanien, A.E. (Eds.). (2019). International conference on innovative computing and communications. International Conference on Innovative Computing and Communications. SpringerLink. <https://doi.org/10.1007/978-981-15-1286-5>
- [64] Schmidt, M., Le Roux, N., Bach, F. (2017). Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162: 83-112. <https://doi.org/10.1007/s10107-016-1030-6>
- [65] Defazio, A., Bach, F., Lacoste-Julien, S. (2014). SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in Neural Information Processing Systems*, 27.
- [66] Johnson, R., Zhang, T. (2013). Accelerating stochastic gradient descent using predictive variance reduction. *Advances in Neural Information Processing Systems*, 1-9.
- [67] Dubois-Taine, B., Vaswani, S., Babanezhad, R., Schmidt, M., Lacoste-Julien, S. (2022). SVRG meets adagrad: Painless variance reduction. *Machine Learning*, 111(12): 4359-4409. <https://doi.org/10.1007/s10994-022-06265-x>
- [68] Zhu, T., Li, G., Zhou, W., Yu, P.S. (Eds.). (2017). Differential Privacy and Applications. SpringerLink. <https://doi.org/10.1007/978-3-319-62004-6>
- [69] Widiyaningtyas, T., Hidayah, I., Adji, T.B. (2022). Comparing user rating-based similarity to user behavior-based similarity in movie recommendation systems. In 2022 International Conference on Electrical and Information Technology (IEIT), Malang, Indonesia, pp. 52-58. <https://doi.org/10.1109/IEIT56384.2022.9967884>
- [70] Nguyen, L.V., Vo, Q.T., Nguyen, T.H. (2023). Adaptive KNN-based extended collaborative filtering recommendation services. *Big Data and Cognitive Computing*, 7(2): 106. <https://doi.org/10.3390/bdcc7020106>
- [71] Cui, L., Huang, W., Yan, Q., Yu, F.R., Wen, Z., Lu, N. (2018). A novel context-aware recommendation algorithm with two-level SVD in social networks. *Future Generation Computer Systems*, 86: 1459-1470. <https://doi.org/10.1016/j.future.2017.07.017>
- [72] Zhang, Z.P., Kudo, Y., Murai, T., Ren, Y.G. (2019). Addressing complete new item cold-start recommendation: A niche item-based collaborative filtering via interrelationship mining. *Applied Sciences*, 9(9): 1894. <https://doi.org/10.3390/app9091894>
- [73] Friedman, A., Berkovsky, S., Kaafar, M.A. (2016). A differential privacy framework for matrix factorization recommender systems. *User Modeling and User-Adapted Interaction*, 26: 425-458. <https://doi.org/10.1007/s11257-016-9177-7>
- [74] Hu, L., Zhang, Y., Wang, Y., Ge, G., Wang, W. (2023). Salient preprocessing: Robotic ICP pose estimation based on SIFT features. *Machines*, 11(2): 157. <https://doi.org/10.3390/machines11020157>
- [75] Lee, M., Hirose, A., Hou, Z., Kil, R.M. (Eds.). (2013). Neural Information Processing. SpringerLink. <https://doi.org/10.1007/978-3-642-42042-9>

- [76] Sun, Y., Lu, T., Yu, Z., Fan, H., Gao, L. (Eds.). (2019). Computer supported cooperative work and social computing. 14th CCF Conference, Chinese CSCW 2019, Kunming, China, August 16–18, 2019, Revised Selected Papers (Vol. 1042). Springer Nature.
- [77] Bobadilla, J., González-Prieto, Á., Ortega, F., Lara-Cabrera, R. (2021). Deep learning feature selection to unhide demographic recommender systems factors. *Neural Computing and Applications*, 33(12): 7291-7308. <https://doi.org/10.1007/s00521-020-05494-2>
- [78] “IOS Press Ebooks - ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August–8 September 2020, Santiago de Compostela, Spain – Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020).” <https://ebooks.iospress.nl/doi/10.3233/FAIA325>.
- [79] Yuan, R., Wang, X., Xu, J., Meng, S. (2021). A differential-privacy-based hybrid collaborative recommendation method with factorization and regression. In 2021 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCCom/CyberSciTech), AB, Canada, pp. 389-396. <https://doi.org/10.1109/DASC-PiCom-CBDCCom-CyberSciTech52372.2021.00073>
- [80] Saleem, F., Iltaf, N., Afzal, H., Shahzad, M. (2019). Using trust in collaborative filtering for recommendations. In 2019 IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), Napoli, Italy, pp. 214-222. <https://doi.org/10.1109/WETICE.2019.00053>
- [81] Hu, M., Liang, H. (2012). Noise-assisted instantaneous coherence analysis of brain connectivity. *Computational Intelligence and Neuroscience*, 2012: 1-1. <https://doi.org/10.1155/2012/275073>.
- [82] Xi, W.D., Huang, L., Wang, C.D., Zheng, Y.Y., Lai, J.H. (2021). Deep rating and review neural network for item recommendation. *IEEE Transactions on Neural Networks and Learning Systems*, 33(11): 6726-6736. <https://doi.org/10.1109/TNNLS.2021.3083264>
- [83] Alexandridis, G., Siolas, G., Stafylopatis, A. (2012). Applying k-separability to collaborative recommender systems. *International Journal on Artificial Intelligence Tools*, 21(1): 1250001. <https://doi.org/10.1142/S0218213012500017>