



## Deep Learning-Driven Multi-Criteria Decision-Making for Effective Recommender Systems

Yahya Bougteb<sup>1\*</sup>    Elyazid Akachar<sup>1</sup>    Brahim Ouhbi<sup>1</sup>    Bouchra Frikh<sup>2</sup>  
 Elmoukhtar Zemmouri<sup>1</sup>

<sup>1</sup>LM2I Lab, National Higher School of Arts and Crafts (ENSAM),  
 Moulay Ismail University, Marjane II, B.P.5290, Meknes, Morocco

<sup>2</sup>LIASSE Lab, National School of Applied Sciences (ENSA),  
 Sidi Mohamed Ben Abdellah University, Atlas, B.P. 1796, Fez, Morocco

\* Corresponding author's Email: y.bougteb@edu.umi.ac.ma

---

**Abstract:** Recommender systems are extensively employed across various domains, encompassing movies, hotels, and restaurants, with the aim of delivering personalized recommendations to users. However, the conventional approach of relying solely on a single rating may fall short in comprehending the underlying factors contributing to a user's overall rating. As a remedy, multi-criteria recommender systems (MCRSs) have emerged as an alternative paradigm, allowing users to rate items based on multiple dimensions. The primary challenge lies in predicting the overall rating considering a user's multi-criteria scores. In this study, we propose a MCRS that leverages a multi-criteria decision-making method, which takes into account the interdependence among criteria. To this end, the weight assigned to each criterion is calculated using the correlation coefficient and standard deviation (CCSD) approach. Furthermore, we employ a deep autoencoder to capture intricate nonlinear relationships and generate recommendations of heightened accuracy. Evaluating our proposed method on the well-known MovieLens and TripAdvisor datasets, we demonstrate its superior performance compared to several single and multicriteria recommender systems. The results highlight the effectiveness of our approach in capturing pertinent aspects and achieving lower prediction errors, showcasing its potential in enhancing recommendation quality and personalization.

**Keywords:** Multi-criteria recommender system, Multi-criteria decision making, Deep learning, Autoencoder.

---

### 1. Introduction

The widespread use of the Internet over the past few decades has resulted in an exponential increase in the amount of available information. This can lead to information overload, which complicates the decision-making process [1]. To address this problem, recommender systems (RSs) have emerged as a promising solution. RSs offer personalized content and assist users in navigating the vast array of options and offers available. By analyzing user input preferences, RSs direct users to the most appropriate information, products, or services, thus aiding them in making informed decisions. For example, in [2], RSs are employed to recommend links in social networks. The approach involves identifying

influential nodes in the network and recommending links to users based on their connections to these nodes. As a result of their effectiveness and practicality, RSs have become a popular research topic and gained significant traction.

Recommender systems that are based on a single criterion (SCRS) make use of user ratings assigned to individual items as the sole evaluation measure. Two primary recommendation algorithm classes exist in SCRS: content-based methods, which rely on the attributes of the items being recommended, and collaborative filtering (CF) based methods, which leverage the preferences and behavior patterns of similar users to produce personalized suggestions. Both of these approaches utilize matrix factorization to learn latent factors for users and items [3–5]. In general, RSs methods can be classified into four main

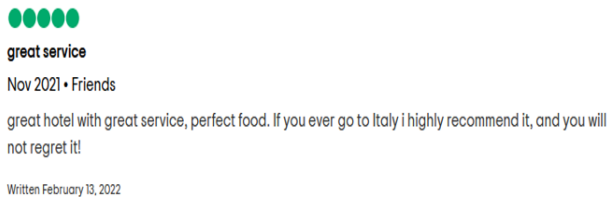


Figure. 1 Snapshot of a hotel review on TripAdvisor website

groups [6]: collaborative filtering, knowledge-based, content-based filtering, and hybrid methods. Less common techniques include demographic methods, community-based methods, and others [7]. A demographic recommender system groups users according to demographic criteria such as nationality, age, gender, and region. In contrast, a community-based recommender system takes into account trust relationships among users.

Traditional RSs use a single criterion rating, known as the overall rating, to generate predictions. This rating reflects a user's overall satisfaction with a particular item [8]. However, relying solely on the overall rating may not be sufficient for predicting user preferences, as it does not provide insight into the underlying reasons for a user's ratings. Consequently, the overall rating may not fully capture the nuances of user preferences or interests in each aspect of an item, making it difficult to analyze user behavior. To address this limitation, multi-criteria rating systems have emerged, which allow users to rate items based on multiple criteria [9]. This approach results in a larger amount of data being collected, enabling the system to provide more appropriate suggestions. The additional criteria also provide more insights into user preferences, which can be useful for improving recommendation accuracy and understanding user behavior [10]. Users often evaluate items based on various criteria, and using only the overall rating for a user-item pair may not be effective. Thus, multi-criteria recommender systems (MCRS) have become popular in the restaurant and travel industries [6]. For example, Zagat's guide rates restaurants using three criteria: service, decor, and food. Buy.com is an online shopping mall that allows multiple criteria for rating consumer electronics, such as battery life, display, performance, and price. TripAdvisor.com is another travel website that provides multiple criteria ratings and allows customers to post opinions and reviews of travel-related content. Fig. 1 shows an example of a typical hotel review written by a user on TripAdvisor. This review comments on the food and service aspects. Based on the review, we can infer that the user values good service and food quality in a hotel,

making these the most relevant aspects for this user in a hotel recommender system. The incorporation of multi-criteria (MC) ratings has been shown to improve recommendation results and increase performance. By allowing users to communicate additional information about their preferences, MCRSs can obtain more information from users and generate more precise recommendations. However, the addition of more criteria can lead to scalability and sparsity issues due to the increase in the rating matrix size. Moreover, in MCRSs, the aggregation of multiple criteria ratings can be a significant challenge [11]. Typically, traditional MCRSs predict an item's overall rating in two stages. First, ratings for each attribute of an item are estimated, followed by the aggregation of those sub-scores using a separate model to predict the overall rating. However, the overall rating heavily relies on the predicted sub-scores of each feature. Therefore, an incorrect prediction of the criteria ratings may significantly affect the overall rating prediction, leading to a decrease in the overall performance of the recommender system.

Recently, deep learning (DL) has shown remarkable success in various fields, including computer vision, natural language processing, and recommender systems. By utilizing DL algorithms, it is possible to extract latent features from raw data effectively, and this approach is becoming increasingly popular in MCRS to tackle issues of sparsity and scalability [12]. Among the DL techniques, the autoencoder, in particular, has gained significant attention for its application in recommendation systems [13, 14]. Autoencoders are neural network architectures that can learn to encode input data into a lower-dimensional latent space and then decode it back to reconstruct the original input. This ability to compress and reconstruct data makes autoencoders well-suited for recommendation tasks. By training an autoencoder on user-item interaction data, it can capture complex patterns and generate latent representations that can be used for personalized recommendations. This approach has shown promising results in improving the accuracy and effectiveness of recommender systems, especially in scenarios with sparse data. However, the existing methods have certain limitations. For example, the AEMC method proposed in [15] uses deep autoencoders to learn the non-linear relationship between users and items but ignores the interdependence among different criteria. Moreover, it relies on the arithmetic mean to calculate the overall rating, which may obscure the variations. Additionally, using a separate deep autoencoder for each criterion slows down the convergence speed

towards the solution. To address these shortcomings, we propose a novel algorithm that incorporates the CCSD approach to accurately compute criteria weights by considering the interdependency between different criteria and the overall rating. To enhance the precision of the system, our approach assigns greater weights to criteria that strongly depend on the overall rating. Moreover, we utilize a deep autoencoder that receives weighted criteria ratings as input to predict the overall rating for each user.

Our contributions are:

- Our first proposal is to adopt a multi-criteria decision-making algorithm that incorporates the CCSD approach [16, 17]. Then, we train a deep autoencoder to predict the overall rating for each user by utilizing the weighted criteria ratings.
- Next, we present four models to illustrate the effectiveness of the DAE-CCSD approach in the domain of MCRS research and the rationale behind it. We compare the DAE-CCSD against these four models, which include approaches that depend on a single evaluation measure, use a single criterion in conjunction with other measures, or adopt the mean value as an aggregation function for multiple criteria.
- Finally, we compare the DAE-CCSD recommendation algorithm against eight well-established baseline methods using the MovieLens 100K and the TripAdvisor datasets. The experimental results demonstrate that the proposed DAE-CCSD method outperforms the baseline methods in addressing the multi-criteria recommendation problem.

The remaining of the paper is organized as follows: Section 2 gives an overview of the related work. Section 3 provides the preliminaries. Section 4 presents the overall framework of our proposed approach. Section 5 discusses the experiments and results. Finally, we end this paper with conclusions and thoughts for future work in section 6.

## 2. Related work

### 2.1 Multi-criteria recommender systems

Multi-criteria rating systems are gaining popularity in recommendation systems as they provide a more detailed and comprehensive understanding of users' references. By allowing users to rate items based on multiple criteria, the system can gather a larger amount of data that can be used to make more accurate and appropriate suggestions. Studies have shown that the relevance of

recommended items for a specific user is dependent on various criteria that the user considers when making a decision [8, 18]. Furthermore, the use of multiple criteria can also help address the issue of the cold start problem, which arises when there is insufficient data to make recommendations for new users or items [19]. Therefore, incorporating multi-criteria rating systems into recommendation systems can lead to more effective and personalized recommendations for users. Thus, several approaches have been proposed to recommend items based on multiple criteria instead of a single criterion. For instance, [20] used multi-criteria ratings and proposed a method to learn and rank the dominating criteria of each item, which were then used to estimate the overall rating using commonly used collaborative filtering methods. Similarly, [21] proposed a multi-criteria framework that incorporated user preferences and opinions on several aspects, with components for rating inference, aspect weighting computing, and opinion mining. The opinion mining component extracted users' opinions from reviews using techniques such as sentiment analysis, and then generated the rating. The second component used a tensor factorization algorithm to infer the weights of distinct aspects automatically and make predictions for the overall rating. In multi-criteria rating systems, users provide explicit ratings for products based on multiple criteria, as well as an overall score. The MCRS rating function can generally be described as follows:  $Users \times Items \rightarrow R_0 \times R_1 \times \dots \times R_k$

where  $R_j$  represents the ratings assigned by users to items according to the  $j$ th criterion, with  $j = 1 \dots k$ , and  $R_0$  denotes the overall rating [8]. To illustrate, a hotel RS typically suggests hotels based on a user's preferences. In a SCRS, a user rates a hotel with a single overall score. However, a MCRS allows users to rate hotels based on various aspects such as price, room, service, and location. In a MCRS, the system uses the feature ratings for each hotel to calculate nearest neighbors, rather than relying solely on overall ratings as in SCRS. This means that users who rate hotels similarly overall may not be considered neighbors in MCRS if their feature ratings differ. The use of features (i.e., criteria) ratings helps the MCRS make more accurate recommendations by taking into account users' specific preferences. In the example shown in Fig. 2,  $U_3$  is predicted to give a rating of 5 for  $H_3$  in MCRS, even though  $U_1$  and  $U_3$  are not considered neighbors in this system because their feature ratings differ, despite having similar overall ratings in SCRS.

MCRS is an effective recommendation system that models users' preferences accurately through







	 H1	 H2	 H3
 U1	3 <sub>5,5,1,1</sub>	5 <sub>2,1,5,5</sub>	2 <sub>2,1,4,4</sub>
 U2	4 <sub>2,2,4,5</sub>	4 <sub>5,5,2,2</sub>	5 <sub>3,3,5,5</sub>
 U3	3 <sub>2,2,5,5</sub>	5 <sub>4,5,2,2</sub>	??

Figure. 2 Illustration of a Hotel MCRS

multi-criteria ratings, as evidenced by several studies [22]. However, the system faces some challenges that need to be addressed, such as determining the weights of criteria [24]. This problem is crucial for MCRS because it needs to understand the interrelationships between criteria to determine the information that is most pertinent to users. This is a multi-attribute decision-making (MADM) problem that deals with decision-making in the presence of multiple criteria. To solve this problem, MCRS commonly employs multi-criteria decision-making (MCDM) methods and multi-dimensional ratings to ascertain user preferences, as discussed in several studies [25–26]. These techniques rely on objective methods to compute criteria weights and determine the overall weight of items to select the best alternative from a finite set of decision alternatives with multiple conflicting criteria. Another approach proposed by [27] is the “Criteria Chains” method. Instead of utilizing all criteria simultaneously, this approach considers a sequential order of criteria. The system computes ratings for each criterion individually, taking into account previous predictions as contextual information. However, calculating the rating for each criterion individually using past predictions as context fails to capture the interdependencies between different criteria at the time when a user makes a decision. Moreover, in their study, [28] employed different similarity metrics and utilized particle swarm optimization to determine the optimal weights for the criteria in a multi-criteria recommender system. Meanwhile, in [29–31], the authors suggested a multi-criteria recommender system that utilizes implicit feedback to enhance their comprehension of the potential of implicit relevance feedback in a multi-criteria rating context. Additionally, to enhance the accuracy of multi-criteria collaborative filtering predictions (MCCF), researchers integrated similarity-based techniques [32], which integrate traditional neighborhood-based methods to solve multi-criteria problems.

## 2.2 Deep learning-based recommender systems

In recent years, the application of deep learning (DL) has expanded greatly and has shown remarkable achievements in various research areas. For instance, in computer vision, DL models have been able to achieve state-of-the-art performance in object detection and recognition tasks [33]. DL has also made significant contributions in natural language processing, such as in question answering and machine translation [34–36]. Additionally, DL has been applied in the field of text mining for topics detection and classification [37]. Furthermore, DL has been used in developing recommender systems, where it has achieved impressive performance in predicting user preferences and providing personalized recommendations [19, 23, 38–40]. In particular, due to the capacity to automatically encode the representation of learnt features and the ability to perform more sophisticated nonlinear transformations. By applying the back-propagation technique during training, the algorithm is able to modify its internal parameters and improve its ability to model complex relationships between input data and output predictions [41]. In a non-linear approach, DL can extract the relationships between users and products [42]. For instance, in [43], deep learning techniques were utilized to extract complex features of items based on their content, and to model and represent non-linear relationships between items and users. The interactions between users, items, and criteria were modeled using a neural factorization machine by the authors of [44].

Autoencoders have become a common deep learning architecture for RSs. They were widely employed in single-criteria recommendation systems [45, 46]. For instance, AutoRec (autoencoder based recommendations) [45] uses an autoencoder to decompose the rating matrix before reconstructing it to predict ratings. The proposed technique uses user or item partial vectors in the input layer to reconstruct them in the output layer. AutoRec can be classified into two types: item-based and user-based. For instance, in the case of item-based AutoRec, given an input  $r^{(i)}$ , the reconstruction is calculated as follows:

$$h(r^{(i)}; \theta) = g(W \cdot f(Vr + \mu) + b) \quad (1)$$

where  $g(\cdot)$  and  $f(\cdot)$  are the activation functions, and  $\theta$  is a set of parameters:  $\theta = \{W, V, \mu, b\}$ , where  $V$  and  $W$  are the weights, and  $\mu$  and  $b$  are the biases.

To tackle the issue of limited single rating RSs, researchers in [47] utilized stacked autoencoders, a deep neural network approach, in a multi-criteria setting. They added an extra layer to the conventional

stacked autoencoders to support multi-criteria ratings, using this additional layer as an input layer. Similarly, the authors in [12] utilized autoencoders for collaborative filtering with multiple criteria, which captures the nonlinear relationships between users and their multi-criteria preferences. In contrast, [46] suggested an autoencoder-based hybrid recommender system that integrates side information. By integrating this information throughout all layers of the network, rather than just the initial layer, the representation generated by dynamic autoencoders can quickly incorporate new data. In [48], a hybrid recommender system utilizing autoencoders, referred to as DHA-RS, was introduced. This technique combines user and item side information through stacked denoising autoencoders with neural collaborative filtering. Neural collaborative filtering is a technique used for predicting user preferences by learning item and user characteristics from additional sources of data, which are commonly referred to as auxiliary data. Lastly, a recommendation system using a convolutional neural network with cross-convolutional filters and global pooling instead of fully connected layers was proposed in [49] to prevent overfitting. In this method, the outer product of features is used to enhance the expressiveness of the model.

To summarize, the studies mentioned above aimed to enhance recommendation effectiveness either by proposing innovative collaborative filtering models that leverage the power of neural networks, or by utilizing neural networks to integrate additional information sources. However, there has been limited research exploring deep learning algorithms for multi-criteria recommendation systems (MCRS) in order to enhance recommendations. Moreover, all the DL-based MCRS methods mentioned earlier focus on acquiring nonlinear latent factors for users and items, but do not consider the interdependence among criteria. They calculate each criterion weight independently of the other criteria. Additionally, the overall rating is often obtained by calculating the average of the obtained criteria rating. However, the average measure is a trend indicator which hides usually the variations in statistics. Therefore, this paper proposes a multi-criteria recommender system based on a MCDM method to calculate each criterion weight allowing to take into account the possible criteria dependency. Then, in order to increase MCRS prediction accuracy, we train a deep autoencoder using all of the weighted criteria ratings.

### 3. Preliminaries

In this section, we present the methods and the

Table 1. Comparative evaluation of the performance of DAE-CCSD against the four proposed models

Algorithm	TripAdvisor		MovieLens	
	RMSE	MAE	RMSE	MAE
MC1-DAE	1.2219	0.9969	1.1759	0.9269
MC2-DAE	1.3953	1.1845	1.2935	1.1145
MC-AVG-DAE	1.1624	0.9012	1.0652	0.8572
SC-DAE	1.5049	1.2227	1.2949	1.1172
DAE-CCSD	<b>0.7944</b>	<b>0.6099</b>	<b>0.6989</b>	<b>0.5425</b>

approaches that will be used throughout this paper, as well as the motivation behind this work.

### 3.1 Motivation

We propose four different models to study the impact of each criterion on the overall rating and to show the DAE-CCSD method's efficiency in the field of MCRS research. Therefore, all the proposed models are either using a criterion and the overall rating, or relying on a single criterion (i.e., the overall rating), or using only the average as an aggregation function of multi-criteria ratings. Thus, we compared DAE-CCSD against the following four models using the MovieLens 100K and the TripAdvisor datasets:

- SC-DAE: a single-criterion (the overall rating) based deep autoencoder.
- MC1-DAE: a deep autoencoder with multiple criteria that utilizes both the primary criterion and overall ratings to enhance its recommendation accuracy.
- MC2-DAE: a deep autoencoder with multiple criteria that utilizes the second criterion alongside the overall rating to improve recommendations.
- MC-AVG-DAE: a deep autoencoder with multiple criteria that uses the average as the aggregation function across multiple criteria to calculate the prediction for the overall rating.

Table 1 presents the results, which demonstrate that all multi-criteria models achieved better performance than the single-criterion model (SC-DAE) on both datasets. In addition, the performance of MC1-DAE was superior to that of MC2-DAE, suggesting that the first criterion carries greater importance to the users compared to the second criterion. However, comparing different criteria can be challenging when dealing with a large number of criteria. Furthermore, the proposed method, namely DAE-CCSD, outperformed the MC-AVG-DAE method. These results suggest that the proposed deep autoencoder, which utilizes the CCSD method to



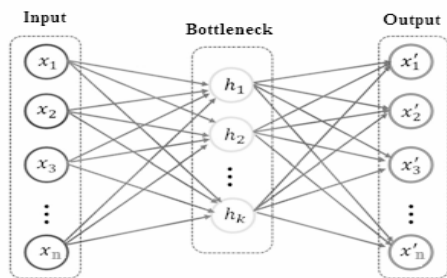


Figure. 3 Autoencoder architecture

compute criteria weights, can effectively capture the correlations among user, item, and criterion dimensions, resulting in improved prediction accuracy.

### 3.2 Autoencoders

Autoencoders are a type of feed-forward neural network that performs encoder and decoder operations (see Fig. 3). They build a dense representation by creating a bottleneck in the neural network architecture before the reconstruction of the model’s original inputs. Furthermore, the autoencoder replaces the traditional linear inner product in matrix factorization methods with a nonlinear decomposition of the rating matrix. During the encoder stage, the input  $X \in R^d$  is transformed into a lower-dimensional encoding  $z \in R^{d'}$  ( $d' < d$ ) through the use of an activation function  $f$ . This transformation is also referred to as the bottleneck.

$$z = f(WX + b) \tag{2}$$

where  $W$  is the weight matrix and  $b$  is the encoding bias vector. In the decoder stage: a vector  $X' \in R^d$  is reconstructed from the dense representation  $z$ :

$$X' = f(W^*z + b^*) \tag{3}$$

where the decoding matrix  $W^*$  and the decoding bias vector  $b^*$  are employed in the decoding process.

### 3.3 Multi-criteria decision analysis

Multi-criteria decision making (MCDM) is a discipline that tries to create tools and procedures for building a trustworthy model. Multi-criteria decision analysis (MCDA) methods address complicated decision-making problems including goals with diverse criteria or multiple choices [50]. An abundance of ways for solving decision-making problems with multiple objectives have been presented and applied in the literature on MCDM. Authors in [51] assume that a decision-making act

maximizes a utility function, which takes into account specific criteria. In circumstances of uncertainty, the problem is usually to maximize the expected value of a utility function. The MCDM methods are designed to assist a decision maker (e.g., an individual or a group) in thinking about the problem during the decision-making process. A complex decision problem can be better understood by breaking it down into five basic components [11]: the situation set, option set, relevant objectives, result function, and preferences for multiple objectives. Therefore, we introduce a new approach based on the CCSD method [16] to pinpoint the key factors that are most significant to the intended user by calculating the weights of each criterion. Thus, the criteria with higher weights will be the most relevant aspects on which the overall rating depends. As result, our recommendation algorithm will focus only on the most relevant aspects and ignores the other attributes that could impact the effectiveness of the suggested recommendations.

## 4. Proposed method

In this section, we present our novel MCRC model, DAE-CCSD, which leverages a deep autoencoder and CCSD method for calculating the different criteria weights. Our approach enables us to develop a recommendation system that identifies the most relevant aspects for the targeted user. To accomplish this, our model takes in the multi-criteria rating histories of users and their respective criterion weights as inputs. First, we calculate the criterion weights to determine their importance, allowing our recommendation system to identify the most relevant aspects for each user. We apply the CCSD method to calculate the weight of each criterion with respect to all users. Next, we train our deep autoencoder to predict each user’s overall rating using all of the weighted criteria ratings as illustrated in Fig. 4. The deep autoencoder algorithm allows for learning high-order features and can deeply learn the nonlinear and hidden relationships between users and items. At this point, we can list and notate the set of items, users, criteria, ratings, and weight of each criterion, as summarized in Table 2.

### 4.1 Deep autoencoder

Autoencoders are feed-forward neural networks that can learn to represent the input data. They are unsupervised networks in which the network’s output attempts to reconstruct the initial input. An autoencoder with increased depth (i.e., more hidden layers) is known as a deep autoencoder. Autoencoders attempt to reduce the mean squared

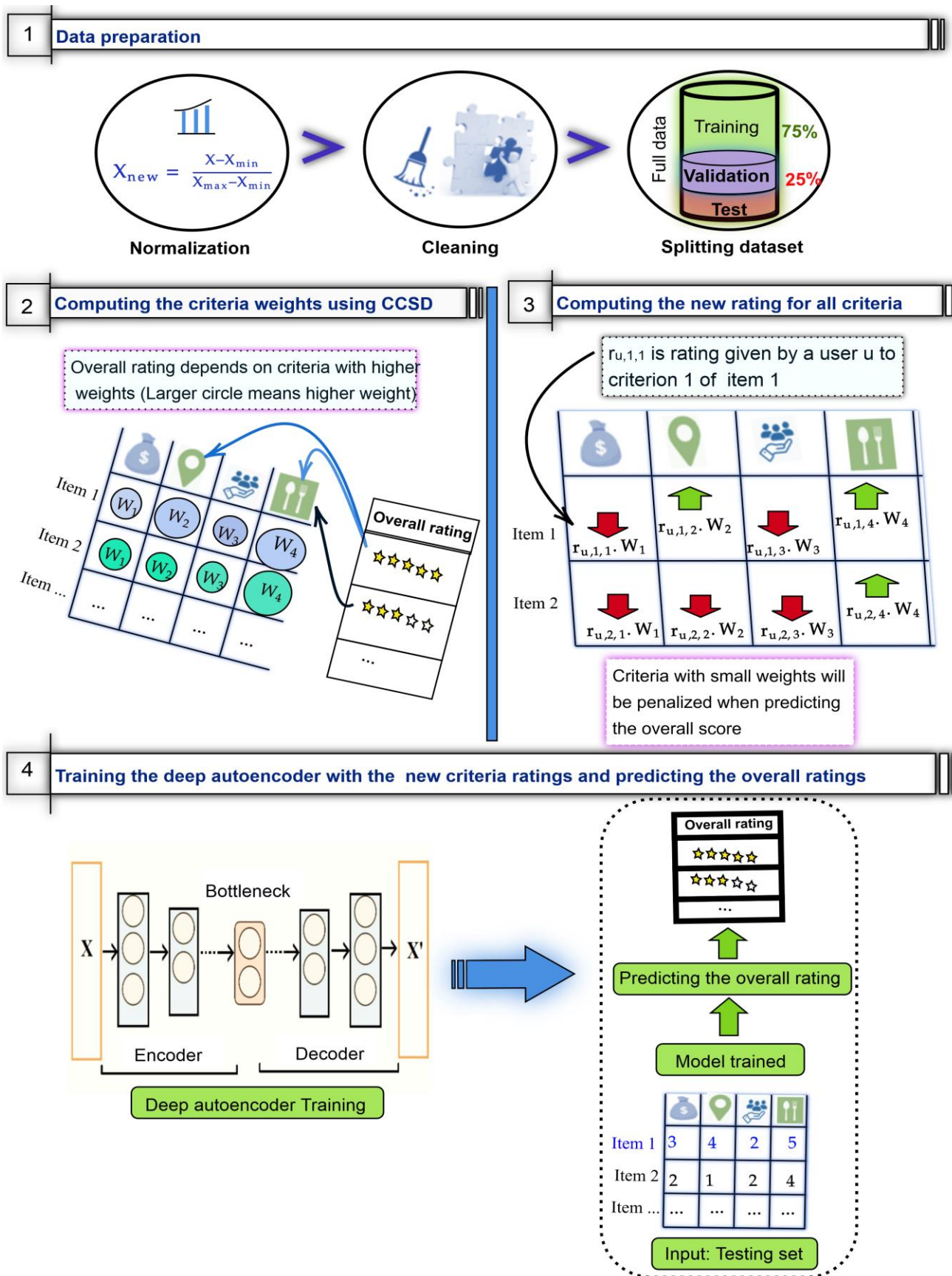


Figure. 4 Overview of our framework

Table 2. Notations

Notation	Description
$U = \{u_1, u_2, u_3, \dots, u_n\}$	Set of $n$ users.
$I = \{i_1, i_2, i_3, \dots, i_m\}$	Set of $m$ items.
$C = \{c_1, c_2, c_3, \dots, c_k\}$	Set of $k$ criteria (i.e., attributes).
$R_{0,u,i}$	The overall rating provided by a user $u$ to item $i$ , considering all relevant criteria.
$\overline{R_{0,i}}$	The mean value of $R_{0,i}$ given by all users to an item $i$ .
$r_{u,i,j}$	The rating given by a user $u$ to criterion $j$ of an item $i$ .
$\overline{r_{i,j}}$	The mean value of $r_{i,j}$ given by all users to criterion $j$ of an item $i$ .
$w_j$	The weight of criterion $c_j$

error (MSE) between the input and output layers as shown in Eq. (4). Hence, they learn the low-dimensional representation (i.e., dense representation) of a matrix.

$$MSE(X, X') = \frac{1}{N} \sum_{i=1}^N (X_i - X'_i)^2 \quad (4)$$

We have developed the DAE-CCSD algorithm, consisting of four main stages. In the first stage, we standardize all the ratings to fit within the interval [0,1] and replace missing values with zeros in the data preparation stage. In the second stage, we calculate the weight of each criterion (i.e., attribute)  $w_j$  with respect to all users using the CCSD method (see next sub-section). Next, in the third stage, we incorporate each obtained weight into the ratings column of the corresponding attribute using the dot product. This approach allows us to adjust the importance of each criterion and assign a weight to each attribute. The new predicted rating  $\hat{r}_{u,i,j}$  given by a user  $u$  to criterion  $j$  is:

$$\hat{r}_{u,i,j} = w_j \cdot r_{u,i,j} \quad (5)$$

where the rating provided by the user for criterion  $j$  of the  $i$ th item is represented as  $r_{u,i,j}$ . Finally, our deep autoencoder is trained on the new predicted ratings  $\hat{r}_{u,i,j}$  and predict the overall rating in the last stage.

The pseudocode of the proposed DAE-CCSD method is summarized in algorithm 1. The activation function in the hidden layers (Eq. (6)) is rectified linear unit (ReLU). Moreover, the sigmoid function is used in the output layer for the overall ratings predictions (Eq. (7)). The regularization term is

employed to avoid overfitting. Therefore, a regularized term is included in the calculation of the reconstruction error to construct the loss function, as shown in Eq. (8). The regularization term is based on the  $l1$  norm. Furthermore, authors showed in [52] that the convergence properties of the Adam optimizer algorithm satisfy the predictions of the theoretical study. Moreover, Adam utilizes the advantages of two popular optimization methods that have emerged recently. Specifically, it leverages the capability of AdaGrad in managing sparse gradients along with the capability of RMSProp in managing non-stationary objectives. Therefore, we applied the Adam optimization algorithm to decrease the loss function's error.

$$ReLU(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases} \quad (6)$$

$$sig(x) = \frac{1}{1+e^{-x}} \quad (7)$$

$$L_{DAE-CCSD} = \sum MSE(X, X') + \lambda \cdot l1 \quad (8)$$

The process of determining the optimal number of neurons in each layer is a critical aspect of every deep learning algorithm. Therefore, we conducted several experiments with varying neuron counts in each layer to identify the most effective configuration for our deep autoencoder. Our primary objective was to minimize MSE of the reconstructed output in comparison to the original input. The formula given in Eq. (9) can be used to calculate the total number of nodes that exist in the hidden layers [53].

$$k = t + \sqrt{n + m} \quad (9)$$

Eq. (9) includes several parameters:  $k$ , which denotes the number of nodes in the hidden layer;  $n$ , which represents the number of nodes in the input layer;  $m$ , which is the number of nodes in the output layer; and  $t$ , a constant value that ranges from 1 to 10.

#### 4.2 Attributes' weights

A branch of statistics called correlation analysis aids in establishing the link between two variables; a high correlation denotes a significant association, while a low correlation denotes a less significant relationship. In this context, a high correlation between a criterion rating and the overall rating denotes a strong dependence between the two. Hence, we should give more weight to that criterion since it will always affect the overall rating. For instance,



**Algorithm 1.** DAE-CCSD algorithm

- 
- 1: **Input:**  
 2:  $X_{u,i,j}$ : multi-criteria rating matrix; // u: user, i: item, j: criterion  
 3:  $\phi$ : set of hyper parameters; //Batch size, learning rate, number of hidden layers...etc.  
 4: **Output:**  
 5:  $R_{0,u,i}$ : the overall rating prediction of user-item ratings;
- 

*Phase 1 – Data preparation*

- 
- 6: **for each** example  $(u,i,j)$  in  $X_{u,i,j}$  **do**  
 7:   **if** ( $r_{u,i,j}$  is missing) then  
 8:      $r_{u,i,j} = 0$   
 9:   **else**  
 10:      $r_{u,i,j} = r_{u,i,j} / 5$ ; // normalization  
 11:   **end if**  
 12:   Split  $X_{u,i,j}$  into a training set (75%) and a test set (25%)  
 13: **end for**
- 

*Phase 2 – Computing the criteria weights*

- 
- 14: **for each** criterion  $j$  in  $C$  **do** //  $C$ : set of  $k$  criteria  
 15: Calculate the Pearson correlation coefficient  $R_j$  between the rating of criterion  $j$  and the overall rating  $R_{0,u,i}$  using Eq. (10)  
 16: Compute its weight  $w_j$  using Eq. (13) and Eq. (14)  
 17: **end for**
- 

*Phase 3 – Computing the new predicted rating for all criteria*

- 
- 18: **for each** example  $(u,i,j)$  in  $X_{u,i,j}$  **do**  
 19: Compute the new predicted rating  $\hat{r}_{u,i,j}$  using Eq. (5)  
 20: // Criteria with small weights will be penalized when predicting the overall score.  
 21: **end for**
- 

*Phase 4 – Training the deep autoencoder and predicting the overall rating*

- 
- 22: // Initialize the hyper parameters  
 23: **for each** epoch **do**  
 24: Encode the input into a dense vector  $z$  in the bottleneck layer using Eq. (2)  
 25: Decode  $z$  and reconstruct the input using Eq. (3)  
 26: Obtain the reconstruction error function MSE according to Eq. (4)  
 27: Add  $l_1$  regularization term to the cost function according to Eq. (8)  
 28: Train the network and update the biases and weights;  
 29: **end for**  
 30: Generate the overall predictions  $R_{0,u,i}$ ;
- 

assume there are  $n$  users  $u_1, \dots, u_n$  with  $m$  choice options  $i_1, \dots, i_m$  that must be assessed in terms of  $k$  criteria (i.e., attributes)  $c_1, \dots, c_k$ . After the user rates items based on multiple criteria, the resulting data is organized in a decision matrix or rating matrix denoted as  $X = (r_{u,i,j})_{m \times n \times k}$ , where  $r_{u,i,j}$  is the rating value assigned by user  $u$  to criterion  $j$  of item  $i$ . In the coefficient correlation and standard deviation approach, the first step involves computing the pearson correlation coefficient  $R_j$  between the criterion  $j$  rating and the overall rating  $R_{0,u,i}$ :

$$R_j = \frac{\sum_{i=1}^m (r_{u,i,j} - \bar{r}_{i,j})(R_{0,u,i} - \bar{R}_{0,i})}{\sqrt{\sum_{i=1}^m (r_{u,i,j} - \bar{r}_{i,j})^2 \sum_{i=1}^m (R_{0,u,i} - \bar{R}_{0,i})^2}} \quad (10)$$

$$R_j = \frac{\sum_{i=1}^m (r_{u,i,j} - \bar{r}_{i,j})(d_{ij} - \bar{d}_j)}{\sqrt{\sum_{i=1}^m (r_{u,i,j} - \bar{r}_{i,j})^2 \sum_{i=1}^m (d_{ij} - \bar{d}_j)^2}} \quad (11)$$

$$d_{ij} = \sum_{l=1, l \neq j}^k x_{il} w_l, i = 1, \dots, m \quad (12)$$

where  $x_{il} = \frac{\sum r_{u,i,l}}{n}$ .

If  $R_j$  is closer to 1 for a criterion  $j$ , this criterion is directly proportional the overall rating. If it is closer to -1, this criterion is inversely proportional to the overall rating. However, if the magnitude of the correlation coefficient is lower or closer to zero, then the criterion rating and the overall rating are probably not strongly dependent on each other. Thus, the criterion  $j$  will have a significant impact on recommendation decision, and it should be given more weight. The weight of each criterion is computed as follows:

$$w_j = \frac{\sigma_j \sqrt{1-R_j}}{\sum_{l=1}^k \sigma_l \sqrt{1-R_l}}, j = 1, 2 \dots k \quad (13)$$

where  $\sigma_j$  is the standard deviation:

$$\sigma_j = \sqrt{\frac{1}{m} \sum_{i=1}^m (r_{u,i,j} - \bar{r}_{i,j})^2}, j = 1, \dots, k \quad (14)$$

Finally, we compute the new predicted rating  $\hat{r}_{u,i,j}$  given by a user  $u$  on the attribute  $j$  of an item  $i$  as shown in Eq. (5).

## 5. Experiments and results

In this section, we present the experimental results and analysis of our proposed DAE-CCSD algorithm, comparing it to several existing single-criterion and multi-criteria recommender systems. The evaluation was conducted on the widely used

Table 3. Datasets description

Dataset	MovieLens 100K	TripAdvisor
Users	1000	3453
Items	1700	1832
Records	100000	21826
Sparsity	94,11%	99.65%
Criteria	Two randomly generated criteria and overall	Value, rooms, location, quality of check-in, cleanliness, service, business services and overall

MovieLens and TripAdvisor datasets, aiming to demonstrate the superiority of our method in terms of prediction accuracy and ranking performance.

### 5.1 Datasets

Experiments are performed on MovieLens<sup>1</sup> 100K [54] and TripAdvisor<sup>2</sup> [55] datasets, as shown in Table 3. MovieLens 100K is a single criterion dataset that includes 100K ratings from 1K users on 1.7K movies. The sparsity level of the dataset was around 94.11 percent. To test the multi-criteria rating approach, we converted the MovieLens 100K dataset by generating two random criteria ratings, each rated on a scale of 1 to 5, for every user. Our goal in conducting experiments on this modified dataset is to assess the efficacy of our proposed approach in a randomized context, where it is challenging for a model to detect and quantify the interdependence between different criteria.

TripAdvisor is a popular travel website that allows users to rate their experiences on a scale of 1 to 5 to rate seven criteria of hotels, including: value for money, the number and quality of its rooms, its location, the quality of check-in, the level of cleanliness, and the availability of service and business amenities. Additionally, users may give the hotel an overall satisfaction rating. Since other academics crawled the website and produced their versions, there is no official dataset for multi-criteria ratings. After cleaning, our dataset contains 21826 records given by 3453 users for 1832 hotels. At least five ratings were given by each user. The dataset's sparsity level was around 99.65 percent.

### 5.2 Evaluation metrics

In order to assess the predictive performance of our proposed method, we employed common statistical measures for accuracy evaluation, such as

the root mean square error (RMSE) and mean absolute error (MAE), which are defined in Eqs. (15) and (16), respectively [56]. We aimed to achieve lower values for both MAE and RMSE, as these would indicate greater accuracy of the predictions.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (r_i - \hat{r}_i)^2} \quad (15)$$

$$MAE = \frac{1}{n} \sum_{i=1}^n |r_i - \hat{r}_i| \quad (16)$$

### 5.3 Compared methods

The DAE-CCSD recommendation algorithm is compared against the following related methods using the MovieLens 100K and the TripAdvisor datasets:

- AEMC [15]: a deep autoencoder based algorithm for multicriteria recommender systems using the arithmetic mean to estimate the overall rating prediction.
- MC-ANN [56]: a multi-layer feedforward artificial neural network model, which consists of two hidden layers.
- MC-UCF [57]: a multi-criteria user-based CF recommendation algorithm.
- SC-UCF: a user-based CF method with a single criterion that uses Constrained Pearson Correlation (CPC) as a similarity measure [15].
- Slopeone [58]: it's the most basic form of item-based collaborative filtering based on ratings.
- SVD: the singular value decomposition algorithm which is a matrix factorization method.
- SVD++: is a matrix factorization model which makes use of implicit feedback information.
- KNN: the k-nearest neighbors' algorithm.

### 5.4 Hidden layers

Based on the results obtained from the experiments outlined in Table 4, it was determined that the number of hidden layers in a deep autoencoder significantly impacts its performance. Specifically, it was observed that deeper networks yielded better outcomes. However, it should be noted that the model began to overfit after the inclusion of more than five hidden layers. This is likely due to the fact that training deep networks requires a significant amount of data. Hence, we selected the deep autoencoder with five hidden layers (referred to as model N<sup>o</sup>3) for our experiments.

<sup>1</sup> <https://grouplens.org/datasets/movielens/latest/>

<sup>2</sup> <http://cs.cmu.edu/~jiweil/html/hotel-review.htm>

Table 4. The effectiveness of deep autoencoder hidden layers

Model	Model 1	Model 2	Model 3	Model 4
#Hidden layers	1	3	5	7
RMSE	1.450	1.402	<b>1.065</b>	1.065
MAE	1.130	1.092	<b>0.857</b>	0.857

Table 5. Performance comparison of DAE-CCSD and other models on MovieLens 100K and TripAdvisor datasets

Model	TripAdvisor		MovieLens	
	RMSE	MAE	RMSE	MAE
SVD	0.9001	0.7171	0.8808	0.7451
SVD++	0.9122	0.7182	0.9110	0.7011
Slopeone	1.0360	0.7959	1.0122	0.7901
KNN with baseline	0.9894	0.7515	0.9562	0.7487
AEMC	0.8956	0.8712	0.7922	0.6311
MC-ANN	1.0260	0.9507	0.9989	0.8807
MC-UCF	1.4499	1.0537	1.2578	1.0017
SC-UCF	1.6515	1.1986	1.4515	1.2486
DAE-CCSD	<b>0.7944</b>	<b>0.6099</b>	<b>0.6989</b>	<b>0.5425</b>

## 5.5 Results and discussion

We divided the dataset into two subsets for the purpose of training our model: a training set and a testing set. The training set was comprised of 75% of the dataset, which was used to assess the precision of the predicted ratings. Additionally, we utilized a Gaussian distribution with a mean of 0 and a standard deviation of 0.01 to initialize the model parameters. Furthermore, we used the adam optimizer method to optimize the model, a learning rate of 0.001 and a batch size of 100. The hidden layers architecture of our DAE-CCSD is  $360 \rightarrow 330 \rightarrow 100 \rightarrow 330 \rightarrow 360$  (i.e., the first hidden layer has 360 nodes, the second has 330 and so on). Table 5 gives the final results.

Table 5 showcases the performance comparison, where all multi-criteria models outperform the single-criterion model (SC-UCF) on both datasets, as expected due to the utilization of more precise information in MCRSs. The integration of the CCSD approach to compute criteria weights, coupled with their incorporation into the MC rating matrix, further improved the accuracy of our proposed method. Among the evaluated methods, matrix factorization techniques (SVD and SVD++) achieved better results compared to the MC user-based collaborative filtering and the MC Artificial Neural Network-based methods. These matrix factorization methods

leverage the extraction of latent factors and introduce regularization and bias terms to minimize RMSE error, optimizing model performance.

MC-ANN is a multi-criteria recommender system that uses artificial neural networks to model relationships between user preferences and item features. While this method allows users to rate items based on multiple criteria, it may face challenges in capturing intricate non-linear relationships within the data, as traditional neural networks often struggle to handle complex data distributions effectively. Furthermore, MC-UCF extends traditional user-based collaborative filtering to handle multiple criteria ratings. Although it aims to provide personalized recommendations based on user similarities across various dimensions, MC-UCF might encounter difficulties when the dataset is highly sparse, as it heavily relies on user-item interactions and may struggle to identify meaningful patterns in sparse data. On the other hand, Slopeone is a traditional collaborative filtering algorithm based on a weighted average approach. While computationally efficient and easy to implement, Slopeone may not fully capture complex user preferences and interdependencies among multiple criteria, limiting its accuracy in handling diverse recommendation scenarios.

The deep autoencoders based multi-criteria algorithms, DAE-CCSD and AEMC, demonstrated superior performance over other methods, capturing intricate non-linear relationships that have predictive value in determining users' preferences, resulting in more accurate recommendations. Importantly, the DAE-CCSD algorithm not only outperforms other deep non-linear models but also showcases its resilience to dataset sparsity. Even with 99.65% sparsity in the TripAdvisor dataset, DAE-CCSD achieved the best results, highlighting its robustness in handling sparse data. Comparing DAE-CCSD and AEMC, DAE-CCSD exhibited the lowest RMSE and MAE scores. The significant performance gain can be attributed to its utilization of the CCSD approach, which assigns more weight to crucial criteria, enabling the model to learn from the most influential factors and enhance overall performance while reducing computation time. In contrast, AEMC, which uses an autoencoder in each criterion dimension, does not sufficiently improve prediction accuracy due to its reliance on the arithmetic mean as the aggregation function and the absence of consideration for criteria dependency.

Our experimental findings demonstrate that the proposed DAE-CCSD model achieved an impressive 11.35% reduction in RMSE and a 7.41% decrease in MAE when evaluated on the largest dataset (i.e.,

MovieLens). These results underscore the effectiveness of our approach in capturing interdependencies between user, item, and criterion dimensions, leading to improved prediction accuracy and enhanced recommendation quality.

## 6. Conclusion

In this paper, we have addressed the challenges posed by recommending items based on multiple criteria, encompassing various aspect ratings in addition to the overall user-item rating. To overcome these challenges, we introduced the DAE-CCSD algorithm, a deep autoencoder-based method that leverages the CCSD approach to calculate criteria weights. By assigning higher weights to the most significant criteria, DAE-CCSD enhances the system's accuracy. Our proposed algorithm incorporates the relationship between the overall rating and other criteria during the training stage, allowing it to learn from the most critical criteria. This approach not only improves performance but also reduces computation time. To evaluate the effectiveness of DAE-CCSD, we conducted extensive experiments and compared it to established baseline methods for multi-criteria recommendation. The results of our experiments clearly demonstrate the superior performance of DAE-CCSD over existing methods. Furthermore, our model outperforms both single-criterion and multi-criteria models on the widely used MovieLens and TripAdvisor datasets.

In future research, we plan to extend our proposed model by incorporating user reviews to account for contextual and semantic aspects. This will further enhance the recommendation quality and personalization capabilities of our system.

## Conflicts of interest

The authors declare no conflict of interest.

## Author contributions

“Conceptualization, Y. Bougteb; methodology, Y. Bougteb, E. Akachar, B. Ouhbi, and B. Frikh; software, Y. Bougteb; validation, B. Ouhbi, B. Frikh, and E. Zemmouri; formal analysis, Y. Bougteb, B. Ouhbi, and B. Frikh; investigation, Y. Bougteb; resources, Y. Bougteb and E. Akachar; data curation, Y. Bougteb and E. Akachar; writing—original draft preparation, Y. Bougteb; writing—review and editing, E. Akachar, B. Ouhbi, B. Frikh, and E. Zemmouri; visualization, Y. Bougteb; supervision, B. Ouhbi and B. Frikh; project administration, B. Ouhbi and B. Frikh; funding acquisition, Y. Bougteb, E.

Akachar, B. Ouhbi, B. Frikh and E. Zemmouri.”

## References

- [1] L. Liu, N. Mehandjiev, and D. L. Xu, “Multi-criteria service recommendation based on user criteria preferences”, In: *Proc. of the fifth ACM Conference on Recommender Systems*, pp. 77–84, 2011.
- [2] G. D. Angelo, L. Severini, and Y. Velaj, “Recommending links through influence maximization”, *Theoretical Computer Science*, Vol. 764, pp. 30–41, 2019.
- [3] J. Wang, “Multi-criteria recommender system with hybrid deep tensor decomposition”, In: *Proc. of 2021 4th International Conference on Data Storage and Data Engineering*, pp. 65–69, 2021.
- [4] V. Faridani, M. Jalali, and M. V. Jahan, “Collaborative filtering-based recommender systems by effective trust”, *International Journal of Data Science and Analytics*, Vol. 3, pp. 297–307, 2017.
- [5] Y. Carmel and B. P. Shamir, “Comparison-based interactive collaborative filtering”, In: *Proc. of Structural Information and Communication Complexity: 22nd International Colloquium, SIROCCO 2015*, Montserrat, Spain, pp. 429–443, 2015.
- [6] D. Monti, G. Rizzo, and M. Morisio, “A systematic literature review of multicriteria recommender systems”, *Artificial Intelligence Review*, Vol. 54, pp. 427–468, 2021.
- [7] R. Burke, “Hybrid web recommender systems”, *The Adaptive Web: Methods and Strategies of Web Personalization*, pp. 377–408, 2007.
- [8] S. M. A. Ghuribi and S. A. M. Noah, “Multi-criteria review-based recommender system—the state of the art”, *IEEE Access*, Vol. 7, pp. 169 446–169 468, 2019.
- [9] K. Lakiotaki, N. F. Matsatsinis, and A. Tsoukias, “Multicriteria user modeling in recommender systems”, *IEEE Intelligent Systems*, Vol. 26, No. 2, pp. 64–76, 2011.
- [10] T. N. T. Tran, A. Felfernig, and N. Tintarev, “Humanized recommender systems: State-of-the-art and research issues”, *ACM Transactions on Interactive Intelligent Systems (TiiS)*, Vol. 11, No. 2, pp. 1–41, 2021.
- [11] H. Dyckhoff and R. Souren, “Integrating multiple criteria decision analysis and production theory for performance evaluation: Framework and review”, *European Journal of Operational Research*, Vol. 297, No. 3, pp. 795–816, 2022.

- [12] A. A. A. Asadi and M. N. Jasim, "Deep learning-based rate prediction model for recommender system using clustering techniques", *International Journal of Intelligent Engineering & Systems*, Vol. 16, No. 2, 2023, doi: 10.22266/ijies2023.0430.38.
- [13] D. Liu, Y. Wang, C. Luo, and J. Ma, "An improved autoencoder for recommendation to alleviate the vanishing gradient problem", *Knowledge-Based Systems*, Vol. 263, p. 110254, 2023.
- [14] A. M. A. Sbou and N. H. Abd Rahim, "An improved hybrid semi-stacked autoencoder for item-features of recommendation system (iHSARS)", *Indonesian Journal of Electrical Engineering and Computer Science*, Vol. 30, No. 1, pp. 481-490, 2023.
- [15] Q. Shambour, "A deep learning based algorithm for multi-criteria recommender systems", *Knowledge-based systems*, Vol. 211, p. 106545, 2021.
- [16] Y. M. Wang and Y. Luo, "Integration of correlations with standard deviations for determining attribute weights in multiple attribute decision making", *Mathematical and Computer Modelling*, Vol. 51, No. 1-2, pp. 1–12, 2010.
- [17] F. Hdioud, B. Frikh, and B. Ouhbi, "Multi-criteria recommender systems based on multi-attribute decision making", In: *Proc. of International Conference on Information Integration and Web-based Applications & Services*, pp. 203–210, 2013.
- [18] D. Jannach, Z. Karakaya, and F. Gedikli, "Accuracy improvements for multi-criteria recommender systems", In: *Proc. of the 13th ACM Conference on Electronic Commerce*, pp. 674–689, 2012.
- [19] Y. Bougteb, B. Ouhbi, B. Frikh, and E. Zemmouri, "A deep autoencoder-based hybrid recommender system", *International Journal of Mobile Computing and Multimedia Communications (IJMCMC)*, Vol. 13, No. 1, pp. 1–19, 2022.
- [20] R. S. Sreepada, B. K. Patra, and A. Hernando, "Multi-criteria recommendations through preference learning", In: *Proc. of the Fourth ACM IKDD Conferences on Data Sciences*, pp. 1–11, 2017.
- [21] C. Yang, X. Yu, Y. Liu, Y. Nie, and Y. Wang, "Collaborative filtering with weighted opinion aspects", *Neurocomputing*, Vol. 210, pp. 185–196, 2016.
- [22] Y. M. Arif and H. N. Hayati, "Learning material selection for metaverse-based mathematics pedagogy media using multi-criteria recommender system", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 6, pp. 541–551, 2022, doi: 10.22266/ijies2022.1231.48.
- [23] M. R. Mundada, "Optimized farming: Crop recommendation system using predictive analytics", *International Journal of Intelligent Engineering & Systems*, Vol. 16, No. 3, 2023, doi: 10.22266/ijies2023.0630.46.
- [24] F. Hdioud, B. Frikh, B. Ouhbi, and I. Khalil, "Multi-criteria recommender systems: A survey and a method to learn new user's profile", *International Journal of Mobile Computing and Multimedia Communications (IJMCMC)*, Vol. 8, No. 4, pp. 20–48, 2017.
- [25] F. Hdioud, B. Frikh, and B. Ouhbi, "Bootstrapping recommender systems based on a multi-criteria decision making approach", In: *Proc. of 2014 International Conference on Next Generation Networks and Services (NGNS)*. *IEEE*, pp. 209–215, 2014.
- [26] H. Ferdaous, F. Bouchra, O. Brahim, M. I. Eddine, and B. Asmaa, "Recommendation using a clustering algorithm based on a hybrid features selection method", *Journal of Intelligent Information Systems*, Vol. 51, No. 1, pp. 183–205, 2018.
- [27] Y. Zheng, "Criteria chains: a novel multi-criteria recommendation approach", In: *Pro. of the 22nd International Conference on Intelligent User Interfaces*, pp. 29–33, 2017.
- [28] P. Choudhary, V. Kant, and P. Dwivedi, "A particle swarm optimization approach to multi criteria recommender system utilizing effective similarity measures", In: *Proc. of the 9th International Conference on Machine Learning and Computing*, pp. 81–85, 2017.
- [29] D. Jannach, L. Lerche, and M. Zanker, "Recommending based on implicit feedback", In: *Proc. of Social Information Access: Systems and Technologies*, Cham, pp. 510-569, 2018.
- [30] M. A. Akcayol, A. Utku, E. Aydoğan, and B. Mutlu, "A weighted multi-attribute-based recommender system using extended user behavior analysis", *Electronic Commerce Research and Applications*, Vol. 28, pp. 86–93, 2018.
- [31] E. R. N. Valdez, D. Quintana, R. G. Crespo, P. Isasi, and E. H. Viedma, "A recommender system based on implicit feedback for selective dissemination of ebooks", *Information Sciences*, Vol. 467, pp. 87–98, 2018.
- [32] N. Manouselis and C. Costopoulou, "Experimental analysis of design choices in



- multiattribute utility collaborative filtering”, *International Journal of Pattern Recognition and Artificial Intelligence*, Vol. 21, No. 02, pp. 311–331, 2007.
- [33] A. Haghighat and A. Sharma, “A computer vision-based deep learning model to detect wrong-way driving using pan-tilt-zoom traffic cameras”, *Computer-Aided Civil and Infrastructure Engineering*, Vol. 38, No. 1, pp. 119–132, 2023.
- [34] A. Kumar, O. Irsoy, P. Ondruska, M. Iyyer, J. Bradbury, I. Gulrajani, V. Zhong, R. Paulus, and R. Socher, “Ask me anything: Dynamic memory networks for natural language processing”, In: *Proc. of International Conference on Machine Learning. PMLR*, pp.1378–1387, 2016.
- [35] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate”, *ArXiv Preprint arXiv:1409.0473*, 2014.
- [36] S. Ranathunga, E. S. A. Lee, M. P. Skenduli, R. Shekhar, M. Alam, and R. Kaur, “Neural machine translation for low-resource languages: A survey”, *ACM Computing Surveys*, Vol. 55, No. 11, pp. 1–37, 2023.
- [37] Y. Bougteb, B. Ouhbi, and B. Frikh, “Deep learning based topics detection”, In: *Proc. of 2019 Third International Conference on Intelligent Computing in Data Sciences (ICDS). IEEE*, pp. 1–7, 2019.
- [38] X. Wang and S. Kadioğlu, “Modeling uncertainty to improve personalized recommendations via bayesian deep learning”, *International Journal of Data Science and Analytics*, pp. 1–11, 2021.
- [39] F. Hdioud, B. Frikh, A. Benghabrit, and B. Ouhbi, “Collaborative filtering with hybrid clustering integrated method to address new item cold-start problem”, In: *Proc. of Intelligent Distributed Computing IX: Proceedings of the 9th International Symposium on Intelligent Distributed Computing–IDC’2015, Guimarães, Portugal*, pp. 285–296, 2016.
- [40] C. Liu, Y. Li, H. Lin, and C. Zhang, “Gnnrec: Gated graph neural network for session-based social recommendation model”, *Journal of Intelligent Information Systems*, pp. 1–20, 2022.
- [41] Y. L. Cun, Y. Bengio, and G. Hinton, “Deep learning”, *Nature*, Vol. 521, No. 7553, pp. 436–444, 2015.
- [42] H. T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, and R. Anil, “Wide & deep learning for recommender systems”, In: *Proc. of the 1st Workshop on Deep Learning for Recommender Systems*, pp. 7–10, 2016.
- [43] Q. Shambour and S. Fraihat, “An item-based multi-criteria collaborative filtering algorithm for personalized recommender systems”, *International Journal of Advanced Computer Science and Applications*, Vol. 7, No. 8, 2016.
- [44] Y. Ding, S. Li, W. Yu, J. Wang, and M. Liu, “A unified neural model for review-based rating prediction by leveraging multi-criteria ratings and review text”, *Cluster Computing*, Vol. 22, pp. 9177–9185, 2019.
- [45] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, “Autorec: Autoencoders meet collaborative filtering”, In: *Proc. of the 24th International Conference on World Wide Web*, pp. 111–112, 2015.
- [46] F. Strub, R. Gaudel, and J. Mary, “Hybrid recommender system based on autoencoders”, In: *Proc. of the 1st Workshop on Deep Learning for Recommender Systems*, pp. 11–16, 2016.
- [47] D. Tallapally, R. S. Sreepada, B. K. Patra, and K. S. Babu, “User preference learning in multi-criteria recommendations using stacked auto encoders”, In: *Proc. of the 12th ACM conference on Recommender Systems*, pp. 475–479, 2018.
- [48] Y. Liu, S. Wang, M. S. Khan, and J. He, “A novel deep hybrid recommender system based on auto-encoder with neural collaborative filtering”, *Big Data Mining and Analytics*, Vol. 1, No. 3, pp. 211–221, 2018.
- [49] S. Lee and D. Kim, “Deep learning based recommender system using cross convolutional filters”, *Information Sciences*, Vol. 592, pp. 112–122, 2022.
- [50] H. Ackermann, A. Newman, H. Röglin, and B. Vöcking, “Decision-making based on approximate and smoothed pareto curves”, *Theoretical Computer Science*, Vol. 378, No. 3, pp. 253–270, 2007.
- [51] J. Wallenius, J. S. Dyer, P. C. Fishburn, R. E. Steuer, S. Zionts, and K. Deb, “Multiple criteria decision making, multiattribute utility theory: Recent accomplishments and what lies ahead”, *Management Science*, Vol. 54, No. 7, pp. 1336–1349, 2008.
- [52] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization”, *ArXiv Preprint arXiv:1412.6980*, 2014.
- [53] Z. Liu, S. Guo, L. Wang, B. Du, and S. Pang, “A multi-objective service composition recommendation method for individualized customer: hybrid mpa-gso-dnn model”, *Computers & Industrial Engineering*, Vol. 128, pp. 122–134, 2019.
- [54] F. M. Harper and J. A. Konstan, “The movielens

- datasets: History and context”, *ACM Transactions on Interactive Intelligent Systems (tiis)*, Vol. 5, No. 4, pp. 1–19, 2015.
- [55] D. Jannach, M. Zanker, and M. Fuchs, “Leveraging multi-criteria customer feedback for satisfaction analysis and improved recommendations”, *Information Technology & Tourism*, Vol. 14, pp. 119–149, 2014.
- [56] S. C. Yücebaşı, “MovieANN: a hybrid approach to movie recommender systems using multi layer artificial neural networks”, *Çanakkale Onsekiz Mart Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, Vol. 5, No. 2, pp. 214–232, 2019.
- [57] G. Adomavicius and Y. Kwon, “New recommendation techniques for multicriteria rating systems”, *IEEE Intelligent Systems*, Vol. 22, No. 3, pp. 48–55, 2007.
- [58] D. Lemire and A. Maclachlan, “Slope one predictors for online rating-based collaborative filtering”, In: *Proc. of the 2005 SIAM International Conference on Data Mining*. SIAM, pp. 471–475, 2005.
- [59] H. Steck, “Embarrassingly shallow autoencoders for sparse data”, In: *Proc. of the World Wide Web Conference*, pp. 3251–3257, 2019.