applied system innovation

Article

Personalized E-Learning Recommender System Based on Autoencoders

Lamyae El Youbi El Idrissi [1,*], Ismail Akharraz [2] and Abdelaziz Ahaitouf [1]

[1] Engineering Sciences Laboratory, Polydisciplinary Faculty of Taza, Sidi Mohamed Ben Abdellah University, Taza 35000, Morocco; abdelaziz.ahaitouf@usmba.ac.ma

[2] Mathematical and Informatics Engineering Laboratory, Faculty of Science Agadir, Ibnou Zohr University, Agadir 80000, Morocco; i.akharraz@uiz.ac.ma

* Correspondence: lamyae.elyoubielidrissi@usmba.ac.ma

Abstract: Through the Internet, learners can access available information on e-learning platforms to facilitate their studies or to acquire new skills. However, finding the right information for their specific needs among the numerous available choices is a tedious task due to information overload. Recommender systems are a good solution to personalize e-learning by proposing useful and relevant information adapted to each learner using a set of techniques and algorithms. Collaborative filtering (CF) is one of the techniques widely used in such systems. However, the high dimensions and sparsity of the data are major problems. Since the concept of deep learning has grown in popularity, various studies have emerged to improve this form of filtering. In this work, we used an autoencoder, which is a powerful model in data dimension reduction, feature extraction and data reconstruction, to learn and predict student preferences in an e-learning recommendation system based on collaborative filtering. Experimental results obtained using the database created by Kulkarni et al. show that this model is more accurate and outperforms models based on K-nearest neighbor (KNN), singular value decomposition (SVD), singular value decomposition plus plus (SVD++) and non-negative matrix factorization (NMF) in terms of the root-mean-square error (RMSE) and mean absolute error (MAE).

Keywords: personalized recommender system; collaborative filtering (CF); e-learning; SVD; SVD++; KNN; NMF; deep learning; autoencoder

check for updates

Citation: El Youbi El Idrissi, L.; Akharraz, I.; Ahaitouf, A. Personalized E-Learning Recommender System Based on Autoencoders. *Appl. Syst. Innov.* **2023**, *6*, 102. https://doi.org/10.3390/asi6060102

Academic Editors: Patrícia Ramos and Jose Manuel Oliveira

Received: 13 August 2023
Revised: 30 September 2023
Accepted: 17 October 2023
Published: 27 October 2023

1. Introduction

Thanks to the rapid development of services offered on the Internet, learners have an increasing number of learning resources available to them in e-learning environments. They can now use computers and mobile devices to search for needed educational information, products and services [1]. It is unnecessary and difficult for a learner to consult all of these learning resources. The integration of recommender systems with e-learning systems allows for filtering and selecting useful and relevant resources for each learner, thus reducing the time needed to choose the right ones. The role of recommender systems is very important when developing an e-learning system to guide learners, hence ensuring a personalized learning environment, called adaptive learning [2].

Adaptive learning is a concept related to personalizing learning for each learner [3,4]. Its objective is to provide each learner with learning activities that are appropriate to their different learning needs [2]. Recommender systems in Adaptive E-Learning (AEL) systems basically focus on recommending relevant and accurate learning resources to the learner.

Recommender systems are one of the information-filtering tools that can predict user preferences for an item [5], reducing the information overload caused by the large volume of information present on the web [6]. These systems have been used in several fields [7–9] and especially in e-commerce to increase turnover by offering relevant articles for each customer [10].

Recommender systems rely on data filtering to recommend items using different recommendation approaches that are generally classified into three categories [8,11,12]: content-based filtering, CF and hybrid filtering.

Regarding content filtering, the system recommends items similar to those that the user likes. To do this, the system builds a profile for the user, including the characteristics of their favorite articles, and then it compares the user's profile with the characteristics of an article to generate recommendations [13,14].

For CF, the system takes into account the preferences of other similar users to the active user and completely ignores item knowledge. To the active user, the system recommends items that other similar users (collaborators or neighbors) have appreciated. To do this, the system calculates the similarity between user preferences and other users' preferences [14,15]. CF approaches take a matrix of ratings (users and items), as shown in Table 1, where each row represents a user and each column represents an item as input, and generally produces the output types based on a predictive value indicating how much the current user likes or does not like an item. The third category is hybrid filtering approaches, which combine the recommendation techniques previously explained to take advantage of their complementary advantages [14,16].

Table 1. Matrix of ratings.

	Item 1	**Item 2**	**Item 3**	**Item 4**	**Item 5**
User1	3		2		3
User2	4	3	4	3	5
User3	3	3		5	4
User4	1	5	5		1

The CF approach is a more common and frequently used approach to design recommender systems [17]. It uses two main methods, memory-based CF and model-based CF. The first method calculates the similarity between users or items by using the existing interactions in the item–user matrix. The second method trains a pretrained model to predict missing ratings by applying machine learning algorithms.

Collaborative model-based approaches use matrix factorization [18], which is a popular approach used in recommender systems to provide personalized recommendations based on user–item interactions. It is a model used to predict ratings, such as SVD, SVD++ and NMF (presented in Section 5.2) and PCA ("Principal Component Analysis"), which is an intelligent approach to analyzing the data structure. PCA produces new variables, "Principal Components (PC)" or latent variables, by maximizing the data variance. PCA applications reduce the dimensions [19].

However, it turns out to be primarily a linear model, which poses the challenge of needing to capture complex nonlinear interactions that may be predictive of user preferences. Despite the known success of traditional collaborative filtering models, they have two main drawbacks, namely, sparsity and cold start, which affect the accuracy of the recommendations [20]. Sparsity is a problem caused by few user ratings. Cold start is a problem that arises for a new user or a new item. The system does not have available information to generate recommendations.

Today, deep learning models have shown significant and efficient results in the fields of speech recognition, image recognition and natural language processing. The strength of deep learning comes from the considerable capacity of neural networks to learn from large datasets with complex patterns. This opens the field to new possibilities for developing recommender systems [14,21].

The integration of deep-learning techniques in recommender systems allows for unveiling a greater capacity to challenge the limits of traditional recommender system techniques and to generate quality recommendations. Recommender system techniques based on deep learning allow better learning of user–item relationships compared to traditional recommender systems [22].

Therefore, to improve the quality of learning and to determine the prediction of learners' preferences for a course in an e-learning environment, we implemented an autoencoder algorithm for collaborative filtering tasks in this study based on the dataset created by Kulkarni et al. [23], and we compared it with four models, KNN, SVD, SVD++ and NMF, using the RMSE and MAE as metrics. The hyperparameters used are "activation function", "optimizer", "batch size", "epoch", "loss function" and "learning rate". This work concerns the use of a database to analyze the performance of a recommendation model based on an autoencoder to recommend appropriate courses to learners.

The rest of this article consists of the following sections: Section 2 presents the main contributions. Section 3 presents the related works concerning recommender systems, and Section 4 describes the methodology used in our study. Section 5 describes the database used, some evaluation metrics and the comparison methods. Section 6 presents the results. Section 7 presents the discussion. Section 8 concludes the article and presents subsequent developments.

2. Motivation and Contributions

The principal contributions of this work are described below:

- A recommendation system based on collaborative filtering was developed to suggest various e-learning courses to learners.
- The system uses a dataset constructed by Kulkarni et al. [23] to analyze the performance of a recommendation model based on an autoencoder to recommend appropriate courses to learners.
- The proposed model was compared to four models: KNN, SVD, SVD++ and NMF.
- MAE and RMSE are the two metrics used to evaluate the performance of these models.

3. Related Work

Several studies on recommender systems have been conducted.

Madani et al. [24] provided a recommendation approach based on social filtering and collaborative filtering for directing learners toward pertinent courses. This approach was developed to determine the optimal way for the learner to learn and to suggest courses that best correspond to the learner's profile and social content.

Kulkarni et al. [6] examined the fundamental paradigms of recommender systems that use explicit and implicit feedback, as well as the many approaches used to create recommender systems to improve learning. They presented a summary of the concepts of e-learning, recommendation systems, and deep learning. In this work, the CF technique was used.

Teodorescu et al. [25] proposed an efficient system for recommending quizzes based on the SVD algorithm, whose aim is to evaluate and show, in real time, the level of knowledge possessed by the learner while using a concept map as part of the course on data structures that are specially created for graph algorithms. The authors compared two groups of learners: the first involved learners who received randomized quizzes, while the second involved learners who received recommended questions. According to the results obtained, the students who were given the recommended questions clearly had an advantage over the others.

Li et al. [26] developed collaborative filtering based on the NMF recommendation algorithm with a privacy protection function for cloud computing, enabling the server to easily collect the data needed for the recommendation while at the same time properly protecting the user's privacy. The experiments demonstrate that the algorithm can reach a certain suggestion accuracy and meet the requirements of a recommendation system based on user privacy protection.

Anwar and Uma [27] proposed an approach that uses collaborative filtering and SVD++ to recommend movies. The suggested method was compared with three models: co-clustering, SVD and KNN. An evaluation of the model using RMSE and MAE showed that collaborative filtering with SVD++ produces fewer errors, with an RMSE of 0.9201 and an MAE of 0.7219. This technique also solves the cold-start and data sparsity problems while providing relevant elements and services.

Zriaa and Amali [28] compared KNN with “k-means clustering” to determine the most efficient method of prediction in an e-learning recommender system. The most commonly employed technique for evaluating the efficiency of an algorithm’s performance in terms of accuracy is mean absolute error (MAE). In addition, the model is better when the MAE is lower.

Al-Nafjan et al. [29] provided a comparison of three models, SVD, SVD++ and NMF, which were examined using “location-based social networks” (LBSNs). The main objective of the developed recommendation system is to predict a user’s restaurant ratings and then generate recommendations based on these predictions. Two performance measures, RMSE and MAE, were used to evaluate this experiment. The SVD approach demonstrated its effectiveness by obtaining the lowest RMSE, while the SVD++ method obtained the lowest MAE measure.

Gomede et al. [30] compared three variants of an autoencoder model—“Collaborative Denoising Auto Encoders (CDAE)”, “Deep Auto Encoders for Collaborative Filtering (DAE-CF)” and “Deep Auto Encoders for Collaborative Filtering using content information (DAE-CI)”—to predict student preferences for learning objects. They proved that the DAE-CF model is the most efficient in terms of adaptability. To compare the obtained results for each model, the authors used “Mean Average Precision (MAPP)”, “Normalized Discounted Cumulative Gain (NDCG)”, “Personalization, Coverage” and “Serendipity” to assess the quality of the recommendations. The dataset used contains the interactions of 3757 students with 5104 “learning objects” of a “Massive Open Online Course” (MOOC) between 2018 and 2019. The interaction is obtained using implicit data (the time spent by a student looking at a learning object).

Sidi and Klein [31] proposed a system for recommending a sequence of questions using the “Neural Collaborative Filtering” (NCF) model, which receives students and questions as input and, as output, produces a series of questions ranked according to their order of difficulty. The proposed model was evaluated using the real-world Algebra1 dataset on four random questionnaires from three different users. The Algebra 1 dataset contains 800,000 attempts to solve a problem by 575 students collected between 2005 and 2006 and showed convincing results, with an average precision correlation (AP) score equal to 0.86.

Q. Zhang et al. [32] presented a course recommendation system using a “Recurrent Neural Network (RNN)” and content-based filtering technique. First, the system takes student enrollment data and course characteristics, and then it uses a content-based filtering algorithm to find similar courses to each course in which students have enrolled, which will be transmitted to the network (RNN) to find the ranking of each course. Learner behavior data from Central China Normal University’s “starC MOOC platform”, with 2142 courses and more than 120,000 users, were used to evaluate the MOOCRC system. The results show that the MOORC model outperforms several traditional recommender algorithms, such as K-nearest neighbor (KNN), content-based recommendation, singular value decomposition (SVD) and restricted Boltzmann machine (RBM) recommender systems, in terms of recommendation accuracy.

Tan et al. [33] developed a deep-learning algorithm named “Attentional Manhattan Siamese Long Short Term Memory (AMSLSTM)” based on the autoencoder to recommend relevant courses to students on MOOC online course platforms. The algorithm takes as input the binary scoring matrix between students and courses. The model was evaluated on the MOOC-Cube Co dataset, consisting of 706 online courses and 199,199 students enrolled in more than 100,000 courses, and was compared with the following algorithms: User-Based Collaborative Filtering (UBCF), Item-Based Collaborative Filtering (IBCF), Bayesian Probabilistic Matrix Factorization (BPRMF) and deep autoencoder network (DeepAE).

H. Zhang et al. [34] implemented a resource recommendation system in online course environments (MOOCRC) based on a Deep Belief Network (DBN). The model receives as input a student's preference matrix for a particular course. Datasets from the "starC MOOC" platform of "Central China Normal University" were used to evaluate the MOOCRC model. The platform has 2142 courses and more than 120,000 users. The results show that the MOOCRC model gives better recommendations compared to several other methods (User_CF, Item_CF, SVD).

Gong and Yao [35] developed a hybrid model that combined a deep collaborative filtering (DeepCF) model with a wide linear model for recommending exercises to students. The DeepCF component uses two stacked denoising autoencoder nets (SDAE) to learn a low dimension of student features and item features. To evaluate the model, the authors collected an online dataset from the education company. They selected 8000 students who took math courses and completed more than 50 h of exercise. The model showed good results, with a relative increase of 10% in the AUC (area under the ROC (receiver operating characteristic) curve) measure compared to the baseline model.

Ren et al. [36] developed an RS called "Neural Collaborative Filtering" (NCF) to predict the grade that a student will receive in a course that he or she plans to take in the next term. This model takes information from students, courses and teachers for the input layer, which is then represented in a latent space in the embedding layer. Then, the layers of the network (NCF) receive the concatenation of the latent vectors of the embedding to finally predict the note. The authors added an activation function (ReLU) to each layer of the network (NCF) to have non-negative values in all layers. The modified NCF model with the nonindicative constraint is called NCFnn. Experiments on a George Mason University dataset showed that the proposed NCF approaches give better results.

X. Pan et al. [1] suggested a deep-learning-based course recommendation approach that produces views from various perspectives and provides course recommendations to students. The implementation of the proposed model is based on the collection of different data types to produce models of students and study programs by examining the relationship between specific models, utilizing deep-learning technology to extract critical features, choosing various recommended features from these actual relationships, and producing various views to make further recommendations to students.

The use of deep-learning algorithms for the development of recommender systems has gradually become widespread in recent years thanks to the progress in research on deep-learning techniques, which has revealed the exceptional performance of these models in terms of feature extraction, predictive classification and feature detection, as previously reported.

Few works in the literature have targeted course recommendation systems, and existing methods have shown their limitations, particularly when working with large-scale and sparse databases. The aim of this study was to develop an autoencoder-based recommender system for predicting learner preferences for courses in an e-learning environment based on collaborative filtering. To this end, we used the database introduced by Kulkarni et al. [23] to create an efficient model for recommending courses to learners. Table 2 briefly summarizes some of the approaches discussed above with the present research.

Table 2. Summary of existing methods with the present study.

Article	Machine Learning Method	Approach	Metric	Dataset	Item Types Recommended
[27]	SVD, SVD++, Co-clustering and K-NN	CF	-MAE -RMSE	MovieLens-100 K	Movie
[29]	SVD, SVD++, NMF	CF	-MAE -RMSE	Yelp dataset	Restaurant

Table 2. *Cont.*

Article	Machine Learning Method	Approach	Metric	Dataset	Item Types Recommended
[30]	Denoising autoencoders, deep autoencoders for collaborative filtering, deep autoencoders for collaborative filtering using content information	CF	-"Mean Average Precision" (MAP) -"Normalized Discounted Cumulative Gain" (NDCG) -"Personalization" (P) -"Coverage" -"Serendipity" (SAUC)	Interactions between students and learning objects from a "Massive Open Online Course" (MOOC)	Learning objects
[31]	Neural Collaborative Filtering (NCF)		-Average precision correlation (AP)	Algebra1 dataset	Question sequencing
[34]	Deep belief networks (DBNs)	CF	RMSE	StarC MOOC platform of Central China Normal University	Course
[35]	Stacked denoising autoencoder (SDAE) with wide linear component	hybrid	Receiver operating characteristic (ROC) curve the area under ROC (AUC-ROC)	Dataset from an online education company	Exercises
Present Approach	Autoencoder	CF	MAERMSE	Dataset created by Kulkarni et al. [23]	Course

4. Methodology and Preliminaries

This section describes the applied preliminaries and the autoencoder algorithm used in this study.

The basic working principle of the autoencoder is to fill in the missing values in the input sparse matrix [30]. This procedure offers two features:

- Learning each student's behavior;
- Predicting the probability of consuming the courses provided. Learning is built based on learner–course interactions, which determine how each learner interacts with the courses presented.

The prediction function refers to the probability, depending on the behavior of each student, of interacting with new courses.

4.1. Problem Definition

The fundamental purpose of personalized recommendations is to predict the user's preferences for items with which he or she has not yet interacted. In other words, it seeks to identify the items most likely to be liked by users based on past rating data. The accuracy of predictions serves as an indicator of system efficiency [37].

In this study, we aimed to predict learner ratings for course recommendations in an e-learning environment using an autoencoder, i.e., predicting unknown ratings for learner–course pairs.

4.2. Autoencoder

Autoencoders are a class of unsupervised neural networks introduced in the late 1980s [38] that aim to learn how to compress and encode input data and then reconstruct the original inputs from the reduced coded representations [39]. The autoencoder is powerful for dimensionality reduction [40].

A basic autoencoder consists of three layers, as shown in Figure 1: the input layer, hidden layer and output layer. The number of neurons in the input layer is equal to the number of neurons in the output layer. This form of neural network is composed of two parts:

an encoder and a decoder. The encoder maps data from the input layer to the hidden layer. The decoder maps the encoded data from the hidden layer to the output layer.

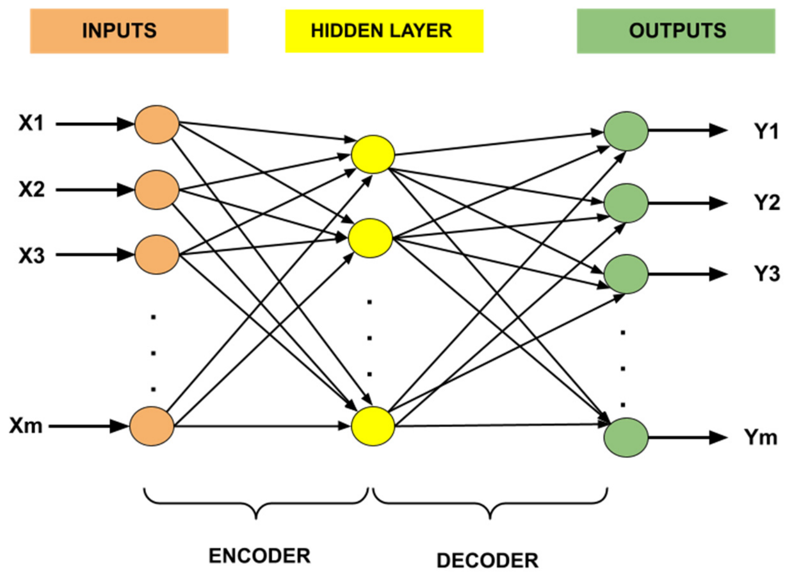

Figure 1. Autoencoder model.

The encoder compresses the high-dimensional input data $x = \{x_1, x_2, x_3, \ldots, x_m\}$ into a latent space representation $h = \{h_1, h_2, h_3, \ldots, h_k\}$ using the function:

$$h = f(x) = S_f(w_{1x} + b_1) \tag{1}$$

where S_f denotes an activation function, m denotes the number of neurons in the input layer, and k denotes the number of neurons in the hidden layer. A $k \times m$ weight matrix W_1 and a bias vector $b_1 \epsilon\ R^k$ are used to parameterize the encoder [38].

The decoder part of the network tends to reconstruct the input from the latent space representation. The reconstruction $y = \{y_1, y_2, y_3, \ldots, y_m\}$ is obtained by using the function:

$$y = g(h) = S_g(w_{2h} + b_2) \tag{2}$$

S_g refers to the activation function of the decoder. An $(m \times k)$ weight matrix W_2 and a bias vector $b_2 \epsilon\ R^m$ compose the decoder's parameters [38].

The purpose of network training is to make the output signal as similar as possible to the input signal. The reconstruction error represents this similarity. The reconstruction error is a loss function that calculates the difference between the original input and the generated output.

Autoencoders are frequently utilized because of their exceptional performance in encoding the original data or learning a representation at the hidden layers. Autoencoders are used for various applications across different fields, such as speech recognition, anomaly detection, computer vision or fault diagnosis [41].

- Configurable Parameters

By reviewing a number of articles on autoencoders in the literature, it was possible to determine that each autoencoder uses a unique set of parameters. It is difficult to choose between them because almost all authors claim that their algorithm or their parameterization method is superior to the others in one way or another. Therefore, the decision was made by trial and error with experimentation using many criteria until the most accurate results were obtained from the model. The hyperparameters used for the model are shown in Table 3.

Table 3. Used hyperparameters.

Hyperparameter	Meaning	Autoencoder
Activation	Function utilized by the neuron's activation	SELU
Batch Size	The size of the sampler that the network is using	64
Epoch	The total number of iterations required for training the network	40
Loss Function	Compares the distance between the prediction output and the target values to determine the model's performance	Mean square error (MSE)
Learning Rate	The rate at which synapse weights are updated	0.0001
Optimizer	"adaptive moment estimation" is an optimization algorithm	Adam

For activation types, there are numerous activation functions. Here are a few examples: "Binary Step", "Hyperbolic Tangent (Tanh)", "Linear Activation", "SoftMax", "Non-Linear", "Sigmoid", "Swish", "Rectified Linear Unit (ReLU)" and "scaled exponential linear unit (SELU)", which is one of the most recent activation functions [30].

Table 4 presents a list of the advantages and drawbacks of some activation functions.

Table 4. Advantages and drawbacks of some activation functions.

Activation Function	Advantages	Drawbacks
Sigmoid	-Simple to understand -Commonly utilized in shallow networks [42]	-Gradient saturation [42] -Slow convergence-Output is nonzero-centered
Tanh	-Output is zero-centered	-Vanishing gradient problem could not be solved using this function [42]
ReLU	-Faster learning	-Fragile during training, resulting in the death of some gradients [42]
SELU	-Not affected by vanishing gradient problems-Works well in standard feed-forward neural networks (FNNs) [43]	-"Internal covariate shift" problem

Using the dataset produced by Kulkarni et al. [23], we carried out experiments comparing the performance of some of the most widely used functions in the literature (notably, SELU, Sigmoid, Relu and Tanh) to select the best activation function for our situation (Table 5).

Table 5. Comparison between SELU, Sigmoid, Relu and Tanh.

Activation Function	MAE	RMSE
SELU	0.6042	0.8756
Sigmoid	1.9906	2.4077
Relu	0.7281	0.9987
Tanh	1.9624	2.3953

From the results in Table 5 for RMSE and MAE (Equations (17) and (18)), SELU improved the RMSE and MAE.

Based on the comparison in Table 4 and the results in Table 5, we chose "the scaled exponential linear unit (SELU)", which has the following function:

$$\text{SELU}(x) = \begin{cases} x & \text{if } (x > 0) \\ \alpha e^x - \alpha & \text{if } (x < 0) \end{cases} \tag{3}$$

Choosing the right optimizer for our model is important for efficient and effective learning. There are several optimization algorithms to consider. Here are a few examples: Stochastic Gradient Descent (SGD) and Adam (Adaptive Moment Estimation).

In our case, we carried out some experiments comparing the performance of Adam and SGD on a dataset made by Kulkarni et al. [23]. Table 6 shows the results of this comparison. It is clearly seen that Adam achieves the best MAE and RMSE results.

Table 6. Comparison between Adam and SGD.

Optimizer Algorithm	MAE	RMSE
Adam	0.6042	0.8756
SGD	1.3769	1.7637

4.3. Procedure for Study

In this section, we present the study plan created to perform a preliminary evaluation of the methodologies employed in the study's later phases. The actions taken are shown in detail in Figure 2.

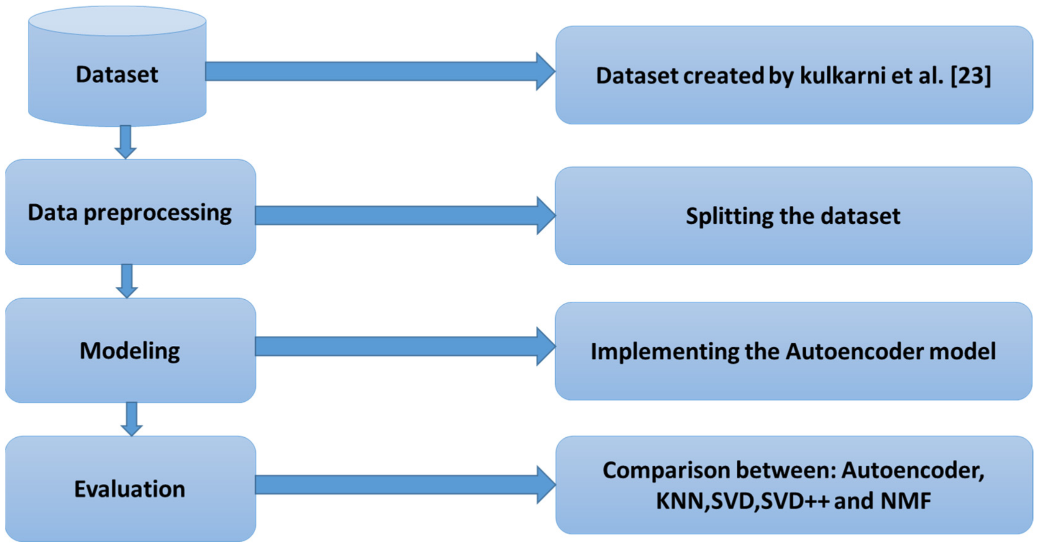
Dataset
Dataset created by kulkarni et al. [23]
Data preprocessing
Splitting the dataset
Modeling
Implementing the Autoencoder model
Evaluation
Comparison between: Autoencoder, KNN,SVD,SVD++ and NMF

Figure 2. Architecture of the study.

In step 1, we selected the dataset created by Kulkarni et al. [23], which contains several files, as described below. In our experiment, we selected the User rating.csv file shown in Table 6 containing student ratings for courses. In step 2, we reorganized the UserId, CourseId and Rating in such a way as to have well-distributed information, while the missing values were already filled with null values, and we split the dataset into 90% training and 10% testing. The autoencoder-based recommendation model was implemented in step 3. Finally, in step 4, we used the RMSE and MAE measures to evaluate the autoencoder with the KNN, SVD, SVD++ and NMF methods.

5. Experiments

In this part, we describe the comprehensive tests that we conducted on the dataset created by Kulkarni et al. [23] to illustrate the efficacy of our proposition.

5.1. The Dataset

The dataset used in this study was published in a work developed by Kulkarni et al. [23] and contains the evaluations of 20 courses carried out by 424 learners and their corresponding profiles.

The statistics on the used dataset are presented in Table 7.

Table 7. Statistics of the dataset.

	Users	Items	Ratings
Dataset	424	20	8480

This dataset includes 3 csv files: User rating.csv, User profile.csv and Master profile.csv:

- User rating.csv contains user ratings and includes user ID, course ID and rating, as shown in Table 8.
- UserId identifies the user. Each user rated courses.
- CourseId identifies the course.
- Rating is the rating ranging from 1 to 5 on a scale of 5 stars.

Table 8. The user rating file with some data.

		CourseId				
		1001	1002	...	1019	1020
UserId	2001	5	3	...	1	3
	2002	3	5	...	0	0
	...	...	...	...	...	...
	2423	2	5	...	5	5
	2424	0	0	...	2	3

User profile.csv contains the profile characteristics of the engineering students presented in Table 9:

- UserId identifies the user.
- Degree 1 is the user's diploma.
- Degree 1 Specializations is the specialty of the user's degree.
- Known languages are languages mastered by the user.
- Key Skills are the skills of the user.
- Career Objective is the career objective of the user.

Table 9. The user profile file with some data.

UserId	**Degree 1**	**Degree 1 Specializations**	**Known Languages**	**Key Skills**	**Career Objective**
1001	B.E.	Computer Science & Engineering	"English, Marathi, Hindi"	C, Java, Keras, Flask, DeepLearning, Selenium, cpp, TensorFlow, Machine Learning, Web Development Areas of interest Django, Python, Computer Vision, HTML, MySQL	"Computer Engineering student with good technical skills and problem solving abilities. include Computer Vision, Deep Learning, Machine Learning, and Research."
1002	B.E.	Computer Science & Engineering	Hindi English	Java, Neural Networks, AI, Python, Html5, CPP	Interested in working under company offering AI/Neural Networking outlooks
...	...	...	...	...	...

Table 9. *Cont.*

UserId	Degree 1	Degree 1 Specializations	Known Languages	Key Skills	Career Objective
2045	B.E.	Computer Science & Engineering		Html, Wordpress, Css, C, Drupal-(CMS) Adobe-Illustrator, HTML, Adobe-Photoshop, MYSQL, Bootstrap, Wordpress-(CMS), JavaScript-(Beginner) Python-(Beginner), CSS	To prove myself dedicated worthful and energetic support in an organization that gives me a scope to apply my knowledge and seeking a challenging position and providing benefits to the company with my performance
2046	B.E.	Computer Science & Engineering		"Python, Robotics", Win32-Sdk, JAVA, Operating-System	"To secure a challenging position where I can effectively contribute my skills as Software Professional, possessing competent Technical Skills."

Master profile.csv contains the characteristics of the profiles of engineering students presented in Table 10, which are:

- UserId, which identifies the user.
- Degree 1, which is the user's diploma.
- Degree 1 Specializations, which is the specialty of the user's degree.
- Campus, which is the name of the campus where the user is registered.
- Key Skills, which are the skills of the user.

Table 10. The master profile file with some data.

Sr	Degree 1	Degree 1 Specializations	Campus	Key Skills
1001	B E	Mechanical,	MITCOE	CATIA
1002	B E	Mechanical,	MITCOE	CATIA
...	...	...	...	...
10,999	B E	Electronics Telecommunication Engineering	MITAOE	"AmazonWebServiCes, C CPP, Arduino, MongoDB, Linux, Golang, Microcontrollers, Gobot, InternetofThings, MATLAB, SQL, PHP"
11,000	B E	Electronics Telecommunication Engineering	MITAOE	"AmazonWebServiCes, C CPP, Arduino, MongoDB, Linux, Golang, Microcontrollers, Gobot, InternetofThings, MATLAB, SQL, PHP"

In our research, we focused on the User rating.csv file.

Figures 3 and 4 show the distribution of ratings for each user and the distribution of course ratings, respectively. We note that the minimum number of ratings assigned by users is 1, while the maximum number of ratings assigned by users is 20. In addition, we see that the minimum number of courses rated by users is 266, and the maximum number of courses rated by users is 393.

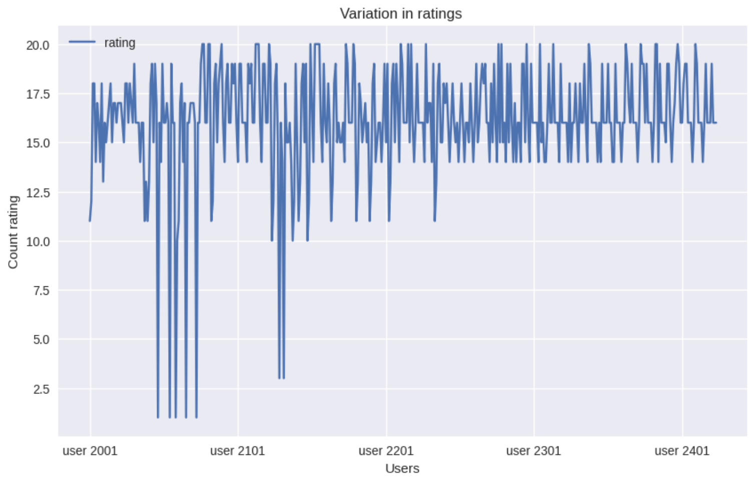

Figure 3. Distribution of ratings for each user.

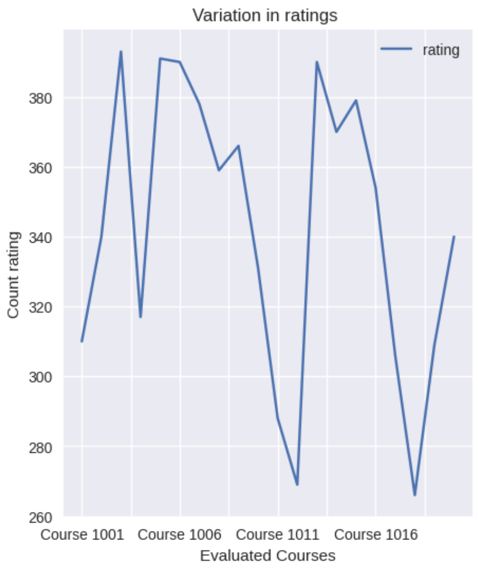

Figure 4. Distribution of course evaluations.

5.2. Compared Methods

We compared our proposed model based on the autoencoder algorithm with traditional methods, including the following:

- KNN

The objective of the collaborative filtering system is to propose elements that are of high interest to a user, either by presenting the predicted rating that a user could give to an element or by providing a probabilistic estimate of the user–element interaction. In other words, the objective of the recommendation system based on collaborative filtering is to determine the top N items for which a user u would give the highest ratings. These items are then proposed to this user. Mathematically, this can be expressed as follows:

$$\text{top}(u, N) = \underset{i \in I}{\overset{N}{\text{argmax}}} \left(\hat{r}_{ui}\right) \tag{4}$$

where:

- I is the list of items that can be recommended.
- N refers to the number of items to recommend.
- $\hat{r}_{ui}$ represents the "prediction of the rating that the recommender system provides to user u for item i".

The fundamental goal of KNN-based CF is to identify K users who share similar behavior to user u and then to suggest items liked by these similar users. The following formula is used to predict user u's rating of item i:

$$\hat{r}_{ui} = \bar{r}_u + \left(\frac{\sum_{v \in U_{ui}^K} \text{sim}(u,v)(r_{vi} - \bar{r}_v)}{\sum_{v \in U_{ui}^K} |\ \text{sim}(u,v)\ |} \right) \tag{5}$$

- U_{ui}^K represents the "K-nearest neighbors" of the user named u who evaluated the item named i.
- r_{vi} denotes the "actual rating given by the neighbor" user v, which concerns item i.
- $\bar{r}_u$ is the average rating relative to user u, which is calculated according to the rating history.
- $\bar{r}_v$ represents the average rating relative to user v, which is calculated according to the rating history.
- $\text{sim}(u,v)$ is the calculation of the similarity between users u and v based on distance metrics, such as cosine and Pearson's correlation coefficient.

However, the value of K in the KNN method is determined before the recommendation procedure. This can be less than optimal, as users have different preferences and similarity levels, resulting in unsatisfactory recommendations. In addition, this method may not be effective for large datasets or when there are few interactions between the user and the item [44].

- SVD

Great dimensionality and data sparsity are common problems in recommender systems. Therefore, dimensionality reduction should be addressed immediately. A powerful method to do this is SVD, which is an MF-specific algorithm [29,45]. According to this SVD approach, an original data matrix can be decomposed into three matrices, shown as:

$$\text{DataMatrix}_{\ m \times n} = U_{m \times m} \Sigma_{m \times n} V^T_{n \times n} \tag{6}$$

$$A = U \Sigma V^T \tag{7}$$

U represents the matrix $m \times m$, Σ is an $m \times n$ matrix with zero elements, except for the values of the diagonal, called "singular values", V is a matrix with $n \times n$ elements and V^T defines the transposed matrix of V.

- SVD++

This technique is a derivative of SVD. The latter uses the information contained in the user–item matrix, which consists of explicit types (ratings or votes), while ignoring implicit information, for example, clicks, purchases and browsing [46].

SVD++ was suggested by [47]. Ref. [48] considers for example a 2D user–item matrix, where p represents rows of users, and q denotes the columns of items. The user rating value for items is $r_{u,i}$. The recommender system aims to predict the items likely to be of interest to users. In fact, the dataset is generally large, and many users can leave items unrated, which poses a data sparsity problem for recommender systems. On the other hand, the "user-item matrix" is decomposed into two further orthogonal matrices by this algorithm: p to represent users and q to correspond to items. The goal is to decompose the initial matrix, which has a large dimension, into smaller data $p_{u,f}$, $q_{i,f}$.

where:

- i is the number of items.
- u is the number of users.
- f denotes the dimension obtained after the reduction in the matrix dimension.

In other words, SVD++ aims to obtain the best representation in low-dimensional space of the "user-item matrix" by deleting irrelevant data and reducing the size of the p and q matrices to a specific size.

The prediction of SVD++ is shown in the following formula:

$$r_{u,i} = \mu + b_u + b_i + q^T\left(p_u + \frac{1}{\sqrt{|N_u|}}\Sigma_{j\in N_u} y_j\right) \tag{8}$$

$$b_i = \frac{\left(\Sigma_{u\in R(i)} r_{u,i} - \mu\right)}{|R_i|} \tag{9}$$

$$b_u = \frac{\left(\Sigma_{u\in R(u)} r_{u,i} - \mu - b_i\right)}{|R_u|} \tag{10}$$

where:

- b_u and b_i are the deviations from the average values for user u and item i, respectively.
- μ represents the average value of all data.
- $|N_u|$ denotes the "number of items" that are assessed by user u.
- $|R_u|$ denotes the "number of users" who have rated a specific item.
- $|R_i|$ represents the "number of items" that have been evaluated by multiple users;
- y_j designates the left orthogonal of implicit matrix.
- $\lambda 1$, $\lambda 2$ are additional parameters added to values $|R_u|$ and $|R_i|$ for regularization [48].

The equations of b_u and b_i can be transformed into:

$$b_i = \frac{\left(\Sigma_{u\in R(i)} r_{u,i} - \mu\right)}{\lambda 1 + |R_i|} \tag{11}$$

$$b_u = \frac{\left(\Sigma_{u\in R(u)} r_{u,i} - \mu - b_i\right)}{\lambda 2 + |R_u|} \tag{12}$$

- NMF

NMF is an unsupervised learning technique that allows dimensionality reduction, and it is based on a lower-rank approximation. A given observation matrix, $X\in R_+{}^{nxm}$, is approximated by NMF as the product of two matrices of non-negative values [49]. In other words, the matrix X can be decomposed into two smaller matrices, W and H, where n indicates the number related to the samples, m is the number that represents the features, k represents the approximation's low rank, $W\in R_+{}^{n\times k}$, $H\in R_+{}^{k\times m}$, $k < m$, $k < n$, and $X \approx W * H$. We achieve this factorization using the multiplicative update approach and non-convex minimization with a non-negativity condition using Frobenius' distance metric, with the following goal function:

$$\min_{W\in R_+^{n\times k},\, H\in R_+^{k\times m}} ||X - WH||_F^2 \tag{13}$$

NMF can thus be viewed as a "Gaussian mixture" model. In Equation (15), the factors W and H represent the optimization problem solution, as determined by alternative updates using the update rules in Equations (14) and (15), respectively, and the performance of the minimization can be evaluated by the "relative reconstruction error" in Equation (16):

$$W = W\frac{XH^T}{WHH^T} \tag{14}$$

$$H = H\left(\frac{W^T X}{W^T W H}\right) \tag{15}$$

$$\text{Relative\_Error} = \frac{\|X - WH\|_F^2}{\|X\|_F^2} \tag{16}$$

5.3. Evaluation Metrics

There are several metrics to evaluate the performance of a recommender system. MAE and RMSE are among the most commonly used [50].

$$\text{MAE} = \frac{\sum_{i=1}^{N}\left(Y_i - \hat{Y}_i\right)}{N} \tag{17}$$

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}\left(Y_i - \hat{Y}_i\right)^2}{N}} \tag{18}$$

where:

- $\hat{Y}$ represents the rating predicted for the user, and Y denotes the original rating of the user;
- N indicates the total number of predicted ratings. Lower values of RMSE and MAE show better prediction accuracy.

5.4. Implementation Details

The experiments were carried out on a Colab with Python 3.10.12 64 bits with Python Machine Learning (ML) Libraries. For the implementation of the autoencoder model, we used TensorFlow 2.13.0 and Keras 2.13.0 deep-learning software, and to create different visualizations, we used the Matplotlib Python library.

6. Results

We ran experiments on the dataset produced by Kulkarni et al. [23] with five models: autoencoder, KNN, SVD, SVD++ and NMF. The full dataset was split into a training dataset subset that contained 90% of the user ratings and another test dataset subset that contained the remaining 10% of the ratings.

Figure 5 displays the model's performance when applied to the dataset. Over the course of each epoch, loss and val_loss decrease until they reach a point where they start to stabilize.

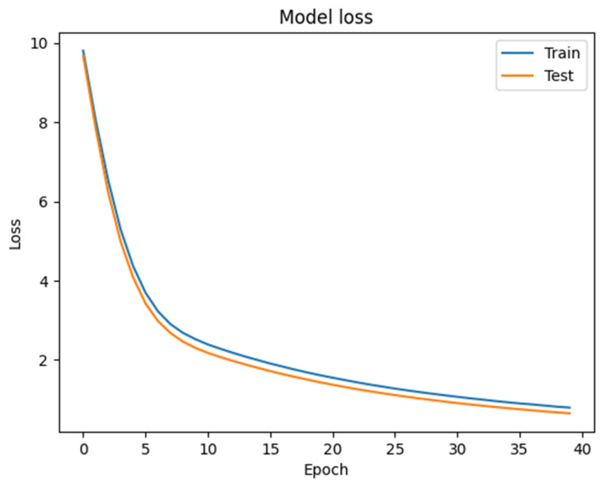

Figure 5. Model loss autoencoder.

The results of the MAE and RMSE comparison between the developed autoencoder model, KNN, SVD, SVD++ and NMF are shown in Table 11 and Figures 6 and 7. The same data processing was performed to produce the KNN, SVD, SVD++ and NMF results using the Surprise module. When we examine these results, we can see that the autoencoder produces better results than KNN, SVD, SVD++ and NMF. The autoencoder model obtained the lowest value of RMSE at 0.8756, the KNN model demonstrated an RMSE of 1.0895, the SVD++ model had an RMSE value of 1.2742, the SVD model had an RMSE of 1.2772 and the NMF model had an RMSE of 1.2851, demonstrating the largest error rate. The MAEs of our developed autoencoder, KNN, SVD, SVD++ and NMF are 0.6042, 0.7259, 0.9922, 0.9796 and 0.9781, respectively. The autoencoder model achieved the best results.

The top-three-recommendation lists produced by the autoencoder model are presented in Table 12.

Table 11. MAE and RMSE comparison of the autoencoder, KNN, SVD, SVD++ and NMF models.

Model	MAE	RMSE
KNN	0.7259	1.0895
SVD	0.9922	1.2772
SVD++	0.9796	1.2742
NMF	0.9781	1.2851
Proposed model (autoencoder)	0.6042	0.8756

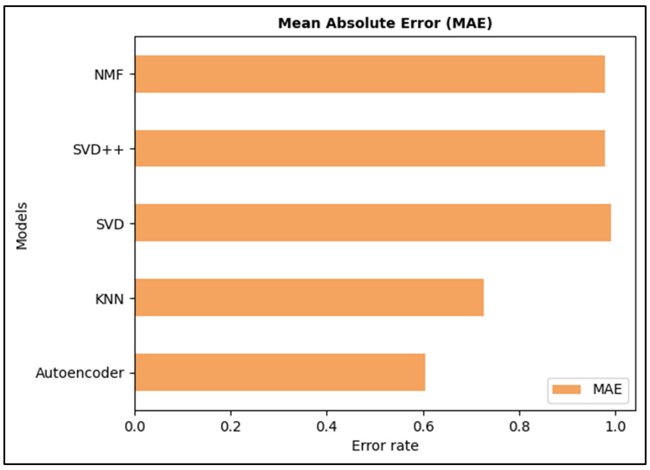

Figure 6. MAE results of different models.

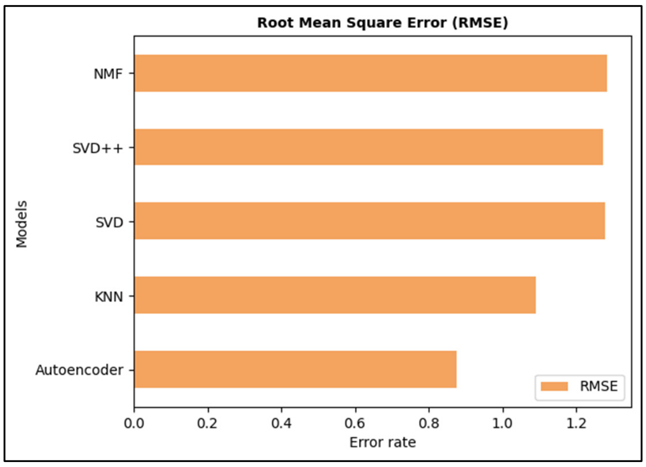

Figure 7. RMSE results of different models.

Table 12. The top three courses recommended with our model for some random users.

UserId	CourseId
2012	[1003, 1006, 1004]
2027	[1016, 1015, 1001]
2141	[1004, 1003, 1005]

7. Discussion

The aim of this work is to develop a personalized course recommendation model based on collaborative filtering in an e-learning environment, and for this purpose, an autoencoder has been used. Experimental results show that the autoencoder has the lowest MAE and RMSE values compared to traditional methods, which are KNN, SVD, SVD++ and NMF. This indicates that the autoencoder model can provide more accurate course rating predictions than the other methods. We can attribute this improvement to the following reasons:

- The autoencoder learns latent representations of user–element interactions, enabling them to capture more complex patterns.
- The autoencoder can handle both dense and sparse data.
- The autoencoder can be more scalable.
- The autoencoder can handle different types of data.

However, the KNN, SVD, SVD++ and NMF models are sensitive to the volume of information available. They perform poorly at high levels of parsimony and are not scalable.

On the other hand, due to the sparsity of standard databases available for e-learning, evaluations in the existing research works in the field of recommending educational resources for e-learning are often based on nonstandard databases or non-accessible databases from various platforms and universities. Moreover, to the best of our knowledge, the dataset created by Kulkarni et al. [23] has never been used to evaluate a course recommendation model. This study aimed to use a reference database to investigate the performance of an autoencoder-based recommendation model to recommend relevant courses to learners. This model can effectively help students in their choice of courses and thus enhance their learning.

8. Conclusions and Future Work

In this paper, we propose a personalized recommendation model in an e-learning environment in which we use an autoencoder-based deep-learning algorithm for course recommendations. Based on collaborative filtering using the dataset developed by Kulkarni et al. [23], the adopted model was used for the prediction of learner ratings of courses. In addition, we compared this approach to KNN, SVD, SVD++ and NMF, which are widely used in recommender systems. We concluded that the autoencoder model is better than the other models for our dataset, with lower MAE and RMSE values.

Autoencoders are efficient algorithms for recommender system tasks, given their ability to capture the main data representations and an effective understanding of the nonlinear relationship between the user and item, user demands and item characteristics, thus solving traditional recommender system problems, such as sparsity and scalability. However, autoencoders present certain drawbacks concerning overfitting when processing a large-scale dataset, the appropriate choice of latent dimensionality, limited ability to handle the cold-start problem for new users and sensitivity to hyperparameters.

This work guides researchers toward adopting the autoencoder algorithm to build a recommendation system in e-learning platforms based on collaborative filtering. As one of the future directions of our work, it would be judicious to use hybrid filtering combined with other deep-learning algorithms, such as generative adversarial networks (GANs) or self-organizing maps (SOMs), to improve the recommendation quality.

Author Contributions: Conceptualization, L.E.Y.E.I., I.A. and A.A.; Methodology, A.A.; Validation, L.E.Y.E.I., I.A. and A.A.; Investigation, L.E.Y.E.I. and A.A.; Resources, I.A.; Writing—original draft, L.E.Y.E.I.; Writing—review & editing, I.A. and A.A.; Visualization, L.E.Y.E.I. and A.A.; Supervision, I.A. and A.A.; Project administration, I.A. and A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the researchers Pradnya Kulkarni, Sunil Rai, Rajneeshkaur Sachdeo and Rohini Kale of "MITADT and Faculty, MITWPU, Pune" for providing us with the reference database [23] that we used in this research, which helped us to evaluate an autoencoder-based course recommendation model for learners.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Pan, X.; Li, X.; Lu, M. A MultiView courses recommendation system based on deep learning. In Proceedings of the 2020 International Conference on Big Data and Informatization Education (ICBDIE), Zhangjiajie, China, 23–25 April 2020; pp. 502–506.
2. Vesin, B.; Mangaroska, K.; Giannakos, M. Learning in smart environments: User-centered design and analytics of an adaptive learning system. *Smart Learn. Environ.* **2018**, *5*, 24. [CrossRef]
3. Normadhi, N.B.A.; Shuib, L.; Nasir, H.N.M.; Bimba, A.; Idris, N.; Balakrishnan, V. Identification of personal traits in adaptive learning environment: Systematic literature review. *Comput. Educ.* **2019**, *130*, 168–190. [CrossRef]
4. Troussas, C.; Sgouropoulou, C. *Innovative Trends in Personalized Software Engineering and Information Systems: The Case of Intelligent and Adaptive e-Learning Systems*; IOS Press: Amsterdam, The Netherland, 2020; Volume 324.
5. Sridevi, M.; Rao, R.R.; Rao, M.V. A survey on recommender system. *Int. J. Comput. Sci. Inf. Secur.* **2016**, *14*, 265.
6. Kulkarni, P.V.; Rai, S.; Kale, R. Recommender System in eLearning: A Survey. In *Proceeding of the International Conference on Computational Science and Applications, Online, 1–4 July 2020*; Springer: Singapore, 2020; pp. 119–126.
7. Lu, J.; Wu, D.; Mao, M.; Wang, W.; Zhang, G. Recommender system application developments: A survey. *Decis. Support Syst.* **2015**, *74*, 12–32. [CrossRef]
8. Guo, Q.; Zhuang, F.; Qin, C.; Zhu, H.; Xie, X.; Xiong, H.; He, Q. A Survey on Knowledge Graph-Based Recommender Systems. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 3549–3568. [CrossRef]
9. Afsar, M.M.; Crump, T.; Far, B. Reinforcement Learning based Recommender Systems: A Survey. *ACM Comput. Surv.* **2022**, *55*, 1–38. [CrossRef]
10. Alamdari, P.M.; Navimipour, N.J.; Hosseinzadeh, M.; Safaei, A.A.; Darwesh, A. A Systematic Study on the Recommender Systems in the E-Commerce. *IEEE Access* **2020**, *8*, 115694–115716. [CrossRef]
11. Singhal, A.; Sinha, P.; Pant, R. Use of Deep Learning in Modern Recommendation System: A Summary of Recent Works. *Int. J. Comput. Appl.* **2017**, *180*, 17–22. [CrossRef]
12. Kumar, P.; Thakur, R.S. Recommendation system techniques and related issues: A survey. *Int. J. Inf. Technol.* **2018**, *10*, 495–501. [CrossRef]
13. Pazzani, M.J.; Billsus, D. Content-Based Recommendation Systems. In *The Adaptive Web: Methods and Strategies of Web Personalization*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 325–341. [CrossRef]
14. Mu, R.; Zeng, X.; Han, L. A Survey of Recommender Systems Based on Deep Learning. *IEEE Access* **2018**, *6*, 69009–69022. [CrossRef]
15. Burke, R. Hybrid Web Recommender Systems. In *The Adaptive Web*; Brusilovsky, P., Kobsa, A., Nejdl, W., Eds.; LNCS 4321; Springer: Berlin/Heidelberg, Germany, 2007; pp. 377–408. [CrossRef]
16. Chen, R.; Hua, Q.; Chang, Y.-S.; Wang, B.; Zhang, L.; Kong, X. A Survey of Collaborative Filtering-Based Recommender Systems: From Traditional Methods to Hybrid Methods Based on Social Networks. *IEEE Access* **2018**, *6*, 64301–64320. [CrossRef]
17. Cui, Z.; Xu, X.; Xue, F.; Cai, X.; Cao, Y.; Zhang, W.; Chen, J. Personalized Recommendation System Based on Collaborative Filtering for IoT Scenarios. *IEEE Trans. Serv. Comput.* **2020**, *13*, 685–695. [CrossRef]
18. Duan, R.; Jiang, C.; Jain, H.K. Combining review-based collaborative filtering and matrix factorization: A solution to rating's sparsity problem. *Decis. Support Syst.* **2022**, *156*, 113748. [CrossRef]
19. Verma, C.; Illés, Z.; Kumar, D. (SDGFI) Student's Demographic and Geographic Feature Identification Using Machine Learning Techniques for Real-Time Automated Web Applications. *Mathematics* **2022**, *10*, 3093. [CrossRef]
20. Alhijawi, B.; Kilani, Y. A collaborative filtering recommender system using genetic algorithm. *Inf. Process. Manag.* **2020**, *57*, 102310. [CrossRef]
21. Wu, L.; He, X.; Wang, X.; Zhang, K.; Wang, M. A Survey on Accuracy-oriented Neural Recommendation: From Collaborative Filtering to Information-rich Recommendation. *IEEE Trans. Knowl. Data Eng.* **2022**, *35*, 4425–4445. [CrossRef]
22. Da'u, A.; Salim, N. Recommendation system based on deep learning methods: A systematic review and new directions. *Artif. Intell. Rev.* **2020**, *53*, 2709–2748. [CrossRef]

23. Kulkarni, P.V.; Rai, S.; Sachdeo, R.; Kale, R. Personalised eLearning Recommendation system. *IEEE DataPort* **2022**. [CrossRef]
24. Madani, Y.; Erritali, M.; Bengourram, J.; Sailhan, F. Social Collaborative Filtering Approach for Recommending Courses in an E-learning Platform. *Procedia Comput. Sci.* **2019**, *151*, 1164–1169. [CrossRef]
25. Teodorescu, O.M.; Popescu, P.S.; Mihaescu, M.C. Taking e-Assessment Quizzes—A Case Study with an SVD Based Recommender System. In *Intelligent Data Engineering and Automated Learning—IDEAL 2018*; Lecture Notes in Computer Science; Yin, H., Camacho, D., Novais, P., Tallón-Ballesteros, A.J., Eds.; Springer International Publishing: Cham, Switzerland, 2018; Volume 11314, pp. 829–837. [CrossRef]
26. Li, T.; Ren, Y.; Ren, Y.; Wang, L.; Wang, L.; Wang, L. NMF-Based Privacy-Preserving Collaborative Filtering on Cloud Computing. In Proceedings of the 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Atlanta, GA, USA, 14–17 July 2019; pp. 476–481. [CrossRef]
27. Anwar, T.; Uma, V. Comparative study of recommender system approaches and movie recommendation using collaborative filtering. *Int. J. Syst. Assur. Eng. Manag.* **2021**, *12*, 426–436. [CrossRef]
28. Zriaa, R.; Amali, S. A Comparative Study between K-Nearest Neighbors and K-Means Clustering Techniques of Collaborative Filtering in e-Learning Environment. In *Innovations in Smart Cities Applications Volume 4. SCA 2020*; Lecture Notes in Networks and Systems; Ahmed, M.B., Kara, İ.R., Santos, D., Sergeyeva, O., Boudhir, A.A., Eds.; Springer International Publishing: Cham, Switzerland, 2021; Volume 183, pp. 268–282. [CrossRef]
29. Al-Nafjan, A.; Alrashoudi, N.; Alrasheed, H. Recommendation System Algorithms on Location-Based Social Networks: Comparative Study. *Information* **2022**, *13*, 188. [CrossRef]
30. Gomede, E.; de Barros, R.M.; de Souza Mendes, L. Deep auto encoders to adaptive E-learning recommender system. *Comput. Educ. Artif. Intell.* **2021**, *2*, 100009. [CrossRef]
31. Sidi, L.; Klein, H. Neural Network-Based Collaborative Filtering for Question Sequencing. *arXiv* **2020**, arXiv:2004.12212. [CrossRef]
32. Zhang, Q.; Li, Y.; Zhang, G.; Lu, J. A recurrent neural network-based recommender system framework and prototype for sequential E-learning. In *Developments of Artificial Intelligence Technologies in Computation and Robotics, Proceedings of the 14th International FLINS Conference (FLINS 2020), Cologne, Germany, 18–21 August 2020*; World Scientific: Singapore, 2020; pp. 488–495.
33. Tan, J.; Chang, L.; Liu, T.; Zhao, X. Attentional Autoencoder for Course Recommendation in MOOC with Course Relevance. In Proceedings of the 2020 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), Chongqing, China, 29–30 October 2020; pp. 190–196.
34. Zhang, H.; Huang, T.; Lv, Z.; Liu, S.; Yang, H. MOOCRC: A Highly Accurate Resource Recommendation Model for Use in MOOC Environments. *Mob. Netw. Appl.* **2019**, *24*, 34–46. [CrossRef]
35. Gong, T.; Yao, X. Deep exercise recommendation model. *Int. J. Model. Optim.* **2019**, *9*, 18–23. [CrossRef]
36. Ren, Z.; Ning, X.; Lan, A.S.; Rangwala, H. Grade Prediction with Neural Collaborative Filtering. In Proceedings of the 2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA), Washington, DC, USA, 5–8 October 2019; pp. 1–10. [CrossRef]
37. Pan, Y.; He, F.; Yu, H. Learning social representations with deep autoencoder for recommender system. *World Wide Web* **2020**, *23*, 2259–2279. [CrossRef]
38. Zhang, G.; Liu, Y.; Jin, X. A survey of autoencoder-based recommender systems. *Front. Comput. Sci.* **2020**, *14*, 430–450. [CrossRef]
39. Ferreira, D.; Silva, S.; Abelha, A.; Machado, J. Recommendation System Using Autoencoders. *Appl. Sci.* **2020**, *10*, 5510. [CrossRef]
40. Kuchaiev, O.; Ginsburg, B. Training Deep AutoEncoders for Collaborative Filtering. *arXiv* **2017**, arXiv:1708.01715.
41. Chen, S.; Guo, W. Auto-Encoders in Deep Learning—A Review with New Perspectives. *Mathematics* **2023**, *11*, 1777. [CrossRef]
42. Nwankpa, C.; Ijomah, W.; Gachagan, A.; Marshall, S. Activation Functions: Comparison of trends in Practice and Research for Deep Learning. *arXiv* **2018**, arXiv:1811.03378.
43. Rasamoelina, A.D.; Adjailia, F.; Sinčák, P. A review of activation function for artificial neural network. In Proceedings of the 2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI), Herlany, Slovakia, 23–25 January 2020; pp. 281–286.
44. Nguyen, L.V.; Vo, Q.-T.; Nguyen, T.-H. Adaptive KNN-Based Extended Collaborative Filtering Recommendation Services. *Big Data Cogn. Comput.* **2023**, *7*, 106. [CrossRef]
45. Chen, V.X.; Tang, T.Y. Incorporating Singular Value Decomposition in User-based Collaborative Filtering Technique for a Movie Recommendation System: A Comparative Study in Proceeding of the 2019 the International Conference on Pattern Recognition and Artificial Intelligence—PRAI'19, Wenzhou, China, 26–28 August 2019; ACM Press: New York, NY, USA, 2019; pp. 12–15.
46. Jiao, J.; Zhang, X.; Li, F.; Wang, Y. A Novel Learning Rate Function and Its Application on the SVD++ Recommendation Algorithm. *IEEE Access* **2019**, *8*, 14112–14122. [CrossRef]
47. Yehuda, K. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, NV, USA, 24–27 August 2008; pp. 426–434.
48. Al Sabaawi, A.; Karacan, H.; Yenice, Y. Two Models Based on Social Relations and SVD++ Method for Recommendation System. *Int. J. Interact. Mob. Technol. (IJIM)* **2021**, *15*, 70. [CrossRef]

49. Eren, M.E.; Richards, L.E.; Bhattarai, M.; Yus, R.; Nicholas, C.; Alexandrov, B.S. FedSPLIT: One-Shot Federated Recommendation System Based on Non-negative Joint Matrix Factorization and Knowledge Distillation. *arXiv* **2022**, arXiv:2205.02359.
50. Zhang, F.; Gong, T.; Lee, V.E.; Zhao, G.; Rong, C.; Qu, G. Fast algorithms to evaluate collaborative filtering recommender systems. *Knowl.-Based Syst.* **2016**, *96*, 96–103. [CrossRef]