IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Ingénierie des Systèmes d'Information
Vol. 29, No. 2, April, 2024, pp. 551-559
Journal homepage: http://iieta.org/journals/isi

Hybrid Recommender System for Personalized Pedagogical Resource Recommendations in E-Learning Platforms

Yassamina Mediani*, Mohamed Gharzouli

MISC Laboratory, Abdelhamid Mehri-Constantine 2 University, Constantine 25016, Algeria

Corresponding Author Email: yassamina.mediani@univ-constantine2.dz

https://doi.org/10.18280/isi.290216

Received: 3 August 2023
Revised: 25 January 2024
Accepted: 25 March 2024
Available online: 25 April 2024

Keywords:
collaborative filtering, deep neural network, E-learning, popularity-based, recommender system, Singular Value Decomposition

ABSTRACT

Recommender systems are generally used in several domains, like e-commerce sites and social networks. E-learning systems use recommendation techniques to facilitate and improve online learning. Educational platforms offer users the necessary pedagogical tools to create an enriched learning environment, fostering collaboration and resource sharing. Recommender System faces many challenges. Among issues: (1) cold-start in which new users and/or items having not prior information available in the system; (2) data sparsity where rated items number is very small contrary to unrated items; and (3) scalability where more training data is required. This study presents a recommender system that uses learner criteria, such as learner's past behavior, demographics information, performance data, collaborative filtering, and ratings to suggest pedagogical resources. The proposed system adopts a hybrid approach, combining two primary methods: popularity-based and collaborative filtering-based. This hybrid approach enhances a collaborative filtering approach with popularity to provide a starting point for new users. The popularity-based is specifically used to address the issue of cold-start for new users by providing primary recommendations. Additionally, we have used two collaborative filtering approaches. The SVD-based enhances the recommendation list for the new user and tackles the sparsity problem. Simultaneously, enhanced matrix factorization with deep neural network (DNN) outperforms traditional matrix factorization in terms of recommendation diversity and accuracy. Our system improves the accuracy and effectively responds to user needs. Our approach early findings show promising results. It scores for top-10 items a total recall of (0.47), a global precision of (0.20), and an accuracy of (0.87).

1. INTRODUCTION

Recommender Systems (RS) are essential software operated within a framework to make sure the management of specific date types online. These RS supply users with overwhelming data and furnish a user a high quality navigation experience. They are widely used in several fields, such as movies, documents, music, and news [1]. Like other platforms, e.g. social services networks and e-commerce, E-learning platforms also try to benefit from recommender systems in order to facilitate and improve learning. These E-learning platforms supply learners with appropriate educational tools and help them construct a better pedagogical environment, including resources sharing and collaboration [2]. Online pedagogical resources are digital entities that involve educational design characteristics, such as online books, articles, and courses [3]. Generally, they are included in E-learning platforms to build an intelligent educational platform.

Recommender systems approaches are arranged into content-based, popularity-based, collaborative filtering-based, and hybrid-based.

Popularity-based recommender systems (PR) are considered a basic form of collaborative filtering (CF), as we will see later, where items shared by most users are considered popular [4]. This system works with trends. Essentially, use currently popular articles. For example, suppose every new user commonly purchases a product. In that case, it can potentially recommend that product to users who have just logged in. Recommending the preferences of most users is somewhat impersonal because it recommends the majority of users' preferences [5].

In Content-based (CB) recommendation systems, the system utilizes the content of items such as categories, descriptions, etc., to construct a profile that includes the extracted information about these items [6]. Typically, the profile consists of all the items liked previously by the user and is automatically updated based on their comments. It uses information search and filtering techniques to match certain candidate items with items that users have previously rated, aiming to recommend the best items to users [7].

In Collaborative Filtering (CF), any given user has mostly similar tastes and preferences to others [8]. The underlying assumption of the collaborative filtering method is that it is possible to make reasonable predictions about the preferences of active users for items they have not yet rated based on the opinions of other users. These methods assume that users with the same preference for one set of items may have the same preference for another that they have not yet rated [9]. CF usually has two main sub-families: model-based and memory-based methods.

Memory-based takes into account all user ratings available when computing recommendations. This approach uses a similarity measure to predict recommendations based on user-wise or item-wise. Where, user-based recommends to users items referring to other like-minded users' opinions on the item. In contrast, item-based collaborative filtering provides a user with an item as a recommendation based on other items with high relevance by calculating their similarity [10].

The fundamental concept behind model-based recommender systems is to construct a "model" using ratings from a dataset. Once user patterns are identified, personalized predictions are automatically generated based on their profiles [10]. The matrix factorization (MF) is the most known model-based approach [11].

To compensate for the weaknesses of each method and provide the best recommendation, a hybrid recommender system (HRS) combines at least two methods. Combining multiple techniques to address the shortcomings of each and exploit their advantages individually has attracted researchers' interest [12]. In general, hybrid filtering uses some strategy to combine the recommendation set to create the final recommendation for the user. Burke identified seven mixing strategies: weighting, cascading, switching, meta-level, feature combination, mixed, and feature augmentation [13]. Among hybrid recommender frameworks having been explored in the context of e-learning, we cite the next online learning platforms:

- LinkedIn Learning which assembles content-based techniques to collaborative filtering to suggest courses. Collaborative filtering determines courses referring to similar users' behavior. Content-based filtering is used to recommend courses that match skills and interests identified from the user's LinkedIn profile and activities [14].
- Coursera which is a massive open online course (MOOC) platform. It uses a hybrid approach to combine collaborative with content-based filtering. Collaborative filtering is used to recommend courses based on very similar students and their behaviors. Content-based methods are used to identify courses that match the user's education, career goals, and implicit interests [15].
- edX which is a MOOC platform. It integrates collaborative filtering with personalized learning paths. Collaborative filtering is used to recommend courses based on the behavior of similar learners. Personalized learning paths are designed using a content-based approach, taking into account the learner's ability level, referred learning style and previous course completion [15].
- Smart Sparrow, which is an adaptive e-learning platform. It combines collaborative filtering with learning analytics. Collaborative filtering recommends courses and activities based on similar learners' preferences. Learning analytics, integrating real-time data on learner interactions and performance, are integrated to dynamically adapt recommendations based on learners' current needs and progress.

In addition to enhancing user navigation and handling large datasets, recommender systems also encounter challenges such as the issue of cold-start, which arises where new users (students in e-learning systems) have no prior information available in the system [16]. Another problem with recommender systems is data sparsity [14], where the rated items number is tiny compared to unrated items. Furthermore, scalability is a major challenge when recommender systems require abundant training data [17].

To enhance the usability and performance of RS, it is important to overcome the issue of cold start. There are many manners to deal with the issue of cold start [18]. Among them:

- Content-Based Recommendations: Those analyze items content and use this information to provide initial recommendations to users with little or no interaction history. This method is efficacious for new items or users because it depend on the items intrinsic properties;
- The use of demographic and contextual information like age, gender, location, etc., to provide initial recommendations to users with limited interaction history;
- Hybrid Models that combine multiple recommendation techniques to take advantage of different approaches. They generally integrate content-based recommendations, demographic information, or other knowledge-based methods with collaborative filtering or matrix factorization. Hybrid models can provide more meaningful recommendations, especially during the cold start phase;
- And finally, the use of popular items to recommend items that are generally popular or highly rated by the entire user base. This helps to provide a starting point for new users.

Our research proposed to assemble collaborative filtering and popularity-based approaches for pedagogical resources by using some hybridization strategies. The popularity-based approach is used with new learners to tackle the issue of cold start by suggesting popular items to new users referring to the user's overall preferences. We use two kinds of CF approaches: Singular Value Decomposition (SVD) and Low-Rank Matrix Factorization (LRMF) enhanced with deep neural networks (DNN) in the CF approach with standard learners.

The SVD is a statistical technique that decomposes one dimension matrix to three matrices, taking the latent factors and the underlying structure of the original matrix. In the recommender systems context, the matrix user-item interaction, with users as rows, items as columns, and entries are user-item interactions (e.g., ratings), is decomposed using SVD. This decomposition helps to identify hidden patterns or hidden features in the data, such as user preferences or item characteristics. So, SVD technique is based on dimensionality reduction and is one of the techniques to generate higher-quality recommendations for users. While, matrix factorization (MF) is a way of decomposing a matrix to a product of two matrices or more, highlighting the latent factors that contribute to the observed data Matrix factorization is used to model user-item interactions. The matrix user-item interaction is decomposed to matrices containing latent features for users and objects. These latent features capture latent patterns, preferences, or characteristics that affect user choice. LRMF enhanced with DNN is a popular technique in recommender systems. It outperforms the traditional MF model and improves the accuracy.

This work proposes our recommender system architecture. It comprises two main modules. The first one represents the separate popularity-based approach, while the second module encapsulates the collaborative filtering approach. Our built system switches between cases based on the learner's criteria. This suggestion addresses the issue of cold start and provides the learner with a list of relevant propositions.

This article is structured as follows. Section 2 presents related work about recommender systems in e-learning. Section 3 presents our proposed architecture. Section 4 discusses data collection, preprocessing, and experiments. Section 5 contains obtained results discussion. Finally, we give a conclusion of our article in section 6.

2. RELATED WORK

Among the several collaborative filtering technologies, matrix factorization techniques play crucial roles to perform the users/items modeling in a similar dimension of users and items victors. The concept of this approach can be represented as Singular Value Decomposition (SVD) [19]. An auto-encoder utilizing a CF method is used by Ferreira et al. [20] to propose a product recommender system. They used the MovieLens1M dataset containing 1 million ratings, and the MovieLens10M dataset containing 10 million ratings. The authors compare the proposed model with the SVD approach in this work. The recommendations offered to users align with their preferences. Results are fairly encouraging; they obtained a Root Mean Square Error (RMSE) of 0.029 in the first dataset and 0.010 in the second. RMSE consists of the quadratic mean of the differences between actual values and the model predictions. The outputs of RMSE are always positive or zero, where zero is considered the best possible value of RMSE.

Furthermore, the problem of cold start is caused by the unavailability of information related to the user or item when he interacts with the system for the first time [17]. In the study by Dadgar and Hamzeh [21], the authors study how social trust improves the performance of the matrix factorization of a recommender system. They propose an asymmetric similarity measure to respect the nature of asymmetrical social relations and link prediction to tackle the cold start problem. Their results on three datasets show that the system performs better in terms of Mean Absolute Error (MAE). MAE consists of the mean of absolute difference between actual values and predictions. The measure is between zero and infinite, where zero is the best possible value.

Like other online platforms, RS are integrated into e-learning environments helping learners in their learning activities. Several types of research and studies about e-learning recommender systems are considered. We can refer to the work of Alqallaf et al. [22], who proposed a hybrid approach combining three recommender frameworks to recommend books appropriate for specific courses. The authors achieved the best F-Measure results of 0.83 by providing recommendations for 100 unrated and relevant books across 13 courses from the BFCI datasets used in their university. Mbaye [23] have proposed a knowledge-based approach using ontology and a decision algorithm for recommending online learning resources to represent the learner's knowledge and learning resources. They supposed their proposition could achieve a better result than similar works.

Mediani [24] have proposed a hybrid RS for pedagogical resources. They used a TF-IDF method to understand better the item's semantics and the SVD method to avoid some weaknesses of the some CF used methods. They used the shared_articles and users_interactions data sets. They have achieved 0.47 for the recall and 0.18 for the precision among top 10 items using SVD based collaborative filtering.

3. THE PROPOSED ARCHITECTURE

The proposed recommender system suggests relevant pedagogical resources that match students' interests. A hybrid approach is developed. It comprised two principal techniques, starting with popularity and then applying SVD-based collaborative filtering. In addition, we have used two hybrid strategies: switching and cascading. In Figure 1, the proposed architecture comprised two modules. The first contains a popularity-based component, while the second module includes the SVD-based CF method.

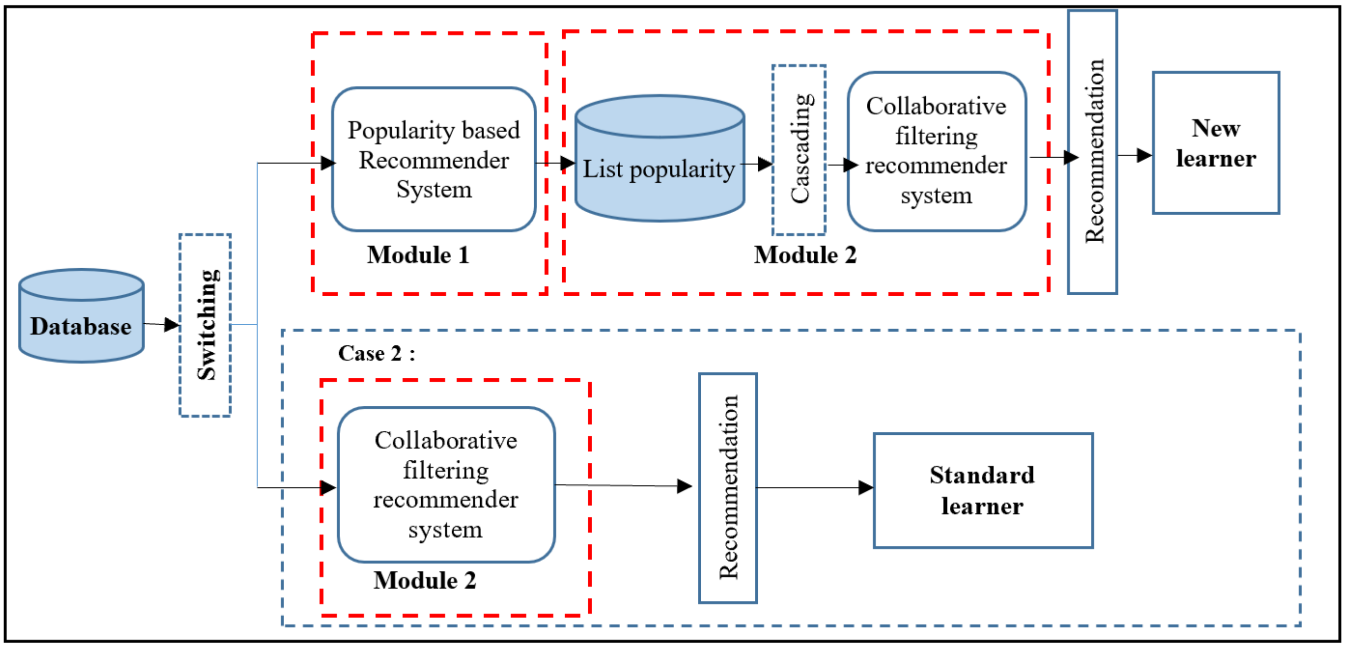

Figure 1. General architecture of our recommender system

Referring to the criteria of the state of learners, this system switches between two cases. The first one is used when there is a newly registered learner to collect information about him. In this case, the final recommended list is based on two phases. The first phase recommends the most popular items and saves the user preferences using the cascading strategy. Then, in the second phase, and to better enhance the recommended list of the first phase, the system uses the SVD-based collaborative filtering to generate other non-popular items based on user preferences. As a result, new learners get both popular and non-popular recommendation items. However, the system switches to the second case when there is a standard learner. In this case, the system already has information about the learner's preferences. Here, the system uses only the second module that contains the SVD-based collaborative filtering approach to provide the recommendation list.

Our architecture is explained as follows:

3.1 Popularity-based recommender system

In this module, we are interested in supplying users with the pedagogical resources which are most popular and having not seen before. We can present the steps of the popularity algorithm as:

Step 1: For every user/item pair, we aggregate the implicit interactions' weights W regarding this item. Because in our system, we have five implicit user interactions (View, Like, Comment, Follow, and Bookmark). To each interaction type is assigned an appropriate weight depending on the level of interaction. For example, Wu, view = 1, Wu, comment = 2.5. In this step, we aggregate all the interactions of the user u regarding a given item i by taking into the count the level of interaction $w_{u,i}$. where, i represent one of the implicit interactions. $W_{u,i}$ represents the user's interest level regarding the item i. It is calculated as follow:

$$W_{u,i} = \sum_{k=1}^{K} \log(1 + w_{u,i}) \tag{1}$$

where, $k \in$ K representing the number of the possible interactions in the system.

Step 2: Collect all the interaction weights regarding each item to find the popular items in the system.

$$tot(W_i) = \sum_{u=1}^{U} W_{u,i} \tag{2}$$

where, $u \in$U represents the number of users interacting with item i. tot(W_i) represents the interest level regarding a given item i of all the users that have interacted with this item. Thus, the items with the higher interaction weight value are considered popular.

Step 3: Generate a candidate list of popular items L1 sorted decently by weight strength.

Step 4: Create new users' primary profiles that contain the generated list L1 and store it in the system.

3.2 Collaborative filtering

In this module, we have adapted two different types of collaborative filtering: SVD collaborative filtering and Low-Rank matrix factorization technique enhanced with DNN architecture.

3.2.1 SVD-based collaborative filtering

We have adapted the SVD matrix factorization technique to address the scalability and data sparsity issues and improve recommendations accuracy. In the recommender system context, SVD decomposes the user/item utility matrix (a sparse matrix) A, defined in the step2 bellow, into three matrices (factors) using equation 3. SVD is used to reduce the utility matrix dimensionality by extracting its latent factors. Each user and item is mapped to an r-dimension latent space. If A initially contains users as rows and resources as columns, then the k-dimensional rows of US contain latent factors for users and those for resources are contained in the columns of SV^T also k-dimensional. After the decomposition of the original matrix A, the dimensionality reduction, and the reconstruction of a new matrix which represent an optimal approximation to the original matrix A by using equation 3., we obtain this optimal approximation without sparsity and scalability.

We can define the SVD steps as follow:

Step 1: For every user/item pair, we aggregate the implicit interactions' weight W regarding this item.

Step 2: The utility matrix A of user/item is built. It contains all U = {u1, u2, …, um} representing users and arranged as rows, and all items I = {i1, i2, …, in} placed as columns. The cells contain the different interactions' weights where 0 is assigned to non-interacted items see Table 1.

Table 1. Example of a user/item utility matrix A

	Item1	**Item2**	**Item3**
User1	4	0	4
User2	0	5	0
User3	1	3	2

Step 3: Break the utility matrix A into its factors with rank(A) =r [25].

$$A = U.S.V^T \tag{3}$$

where, $U_{m,r}$, $V_{r,n}$ are orthogonal matrices whose columns represent the eigenvectors of $A.A^T$ and $A^T.A$, respectively.

$S_{r,r}$ is the diagonal matrix with r non-zero elements. Where: ($\partial 1$, $\partial 2$, …, ∂r) are the r initial diagonal elements of S with $\partial 1 \geq \partial 2 \geq ... \geq \partial k > 0$.

Step 4: We perform the matrix factorization of A by providing the optimal approximation to the matrix A, and by applying the multiplication of the three smaller matrices [25].

$$U_k, S_k, V_k^T = SVD\_function(A, r) \tag{4}$$

In the matrix S, we keep the largest k singular values, where first k rows and first k columns of S are set equal to S_k of (k × k matrix). In addition, we keep the largest k right singular vectors, where first k rows and first k columns of V^T are set equal to V_k^T of (k × n matrix). Finally, first k rows and first k columns of U are set equal to U_k of (m× k dimension) to keep the k left singular victor [25].

Step 5: Get and sort the user's predictions.

Step 6: Recommend the pedagogical resources with the highest predicted ratings that the user has not yet seen.

Algorithm 1: SVD (Singular Value Decomposition)

```
Input: A: matrix with m*n, number k
Output: Approximate $U_k$, $S_k$, and $V_k$
Generate a Gaussian matrix G of n*k.
```

```
Calculate Y=A*G.
Calculate an orthogonal column base Q of Y.
Form B = $Q^T$A
Calculate the proper decomposition of $BB^T = AS^2A^T$.
$U_K$= QA, $V_K = B^T A\,S^{-1}$ et $S_K$ = S.
```

3.2.2 LRMF enhanced with DNN architecture

This module employs a LRMF technique enhanced with DNN architecture to ameliorate recommendation accuracy. We have built the model by combining the MF linearity and the non-linearity of the multi-layer perceptron in order to model the latent structure of user elements. The algorithm and architecture are explained below:

Step 1: For every user/item pair, we aggregate implicit interactions' weights W regarding this item.

Step 2: Create the utility matrix y_{ui} of user-item, which comprises the users U = {u1, u2 …um} placed as rows and the items I = {i1, i2 …in} placed as columns. This obtained matrix is very sparse, with non-interactive elements assigned as 0.

Step 3: Divide the utility matrix into two latent factors which are low-rank, the factor p_u of the user, and the factor q_i of the item [26]:

$$\widehat{y_{ui}} = f\{u, i|p_u, q_i\} = \sum_{k=1}^{K}(p_{uk} \times q_{ki}) \qquad (5)$$

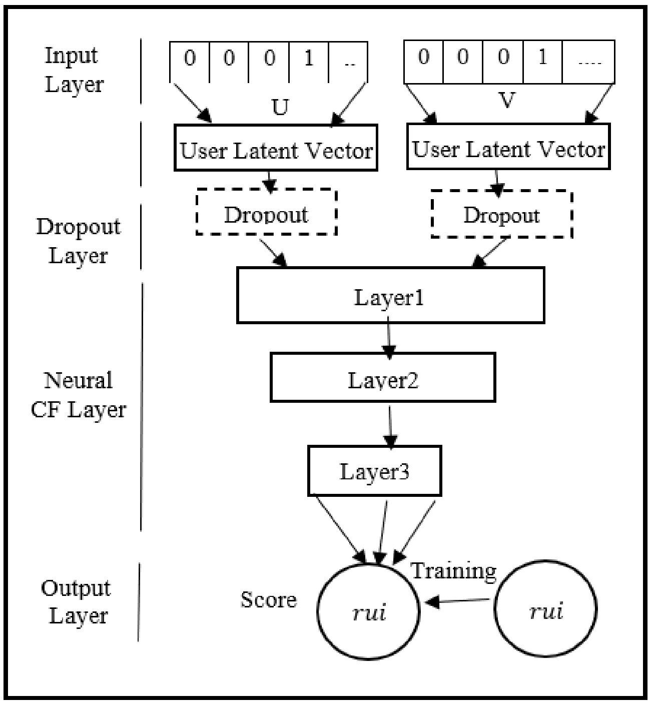

Figure 2. Neural low-rank MF architecture

Step 4: Construct the DNN architecture presented in Figure 2 [27]:

- The first network layer is the input layer, with two input vectors v_u, and v_i, representing a user u and an item i, respectively. These are sparse real-value vectors of user-item pair W_{ui} interaction weights.
- Apply dropout operation to the input layer to simulate the low-ranking of low-rank matrix factorization by randomly deleting some Historical removal interaction.
- Build an embedding layer. This completely connected layer (with some dropout) mainly projects sparse representations into dense vectors. As a result, we have user-item latent vectors.

$$\widehat{y_{ui}} = f\{P^T.v_u, Q^T.v_i|P, Q, \theta_f\} \qquad (6)$$

where, P∈ $R^{m\times k}$ and Q∈ $R^{k\times n}$ represent users and items compressed latent-factor matrix.

θ_f represents the hyperparameters of the interaction function model. f is a multi-layer neural network that is formulated as follows:

$$f\{P^T.v_u, Q^T.v_i|P, Q, \theta_f\} = \emptyset_{out}(\emptyset_X(\ldots\emptyset_2(\emptyset_1(P^T.v_u, Q^T.v_i))\ldots)) \qquad (7)$$

- To predict scores, we feed the embedding layers into a neural architecture of multi-layer with varied number of neurons and hidden layers to find new latent structures from the interactions user-item. We combine the user and item paths to construct a proficient deep learning-based recommender system. However, capturing interactions between latent features of items and users while the concatenation of a simple vector is not enough. So we tackled this issue by adding hidden layers into the combined vector and used multi-layer perceptron (MLP) to acquire how latent vectors of the user and item interact.
- Construct the final layer that gives the predicted interaction score $\widehat{y_{ui}}$ (implement Relu activation function).
- Train the network model, where the objective is minimizing the pointwise loss of ($\widehat{y_{ui}}$, y_{ui}) representing the predicted and the actual value respectively.

Step 5: For every user, we generate a ranked candidate list of items according to the predictions given by the network.

4. EXPERIMENTS

Several experiments have been done to prove the effectiveness and efficiency of our method. We have used Python 3 to implement them on Jupyter Notebook on Windows 10 (64 bits) with an Intel corei5 processor and 8 GB memory. Our built educational recommender system must be answered these questions:

- Does the proposed system provide a recommendation for new students and prevent user cold start problems?
- Does this system provide a recommendation to the users with high accuracy?

4.1 Datasets

We choose articles to represent the pedagogical resources for a better train and test of the proposed RS and validate the proposed approach to build this system. For that, we have used a dataset that is twelve months samples (March 2016 - February 2017) collected from Desk Drop platform. It includes two files:

- The shared_resources.csv dataset is composed of 3128 samples collected from shared articles. The samples contain information about these articles such as date (timestamp), article URL, title, text (content), article language, and author information.

- The users_interactions.csv dataset comprises 72312 samples collected from the user interactions logs on the shared articles. We join the shared_resources.csv by the column "contentId". This dataset contains eventType composed of some values such as view, like, comment, follow, and the bookmark (the article bookmarked are saved to make it easier to access in the future).

4.2 Dataset pre-processing

To know the user preferences, it is necessary to extract what the user is interacting with. Moreover, since the data contains implicit user interactions, it is essential to associate a weight to this interaction to feed it to the machine learning algorithms. These associated weights referred to the level of user interactions. We set the weight from one to five for each user interaction (view, like, bookmark, follow, and comment), respectively.

For data engineering, we have dropped the unnecessary features and kept only the features used by our recommender system (personId, contentId, Title, Lang, interaction_weight). We have also removed the duplication by eliminating duplicate copies of repeating interactions stored in the dataset. In addition, instead of eliminating articles with non-English language, we have translated the Portuguese article language to English using an API for translation to unify and obtain more data for a good training of the model.

4.3 Evaluation metrics

The recommended list of pedagogical resources must be relevant and provide more accuracy. Multiple metrics are utilized to determine the quality of our RS, like precision, recall, accuracy, etc.

In this experiment, we have applied accuracy, f1_score, precision, and recall to assess the system's performance, which gives a top_N recommendations list containing the most interesting resources to a given user. The resources could be recommended or not recommended, relevant or not relevant. These metrics are defined as follows [28]:

Recall@N: indicates the ratio of the relevant items existing in Top_N recommendations.

$$recall_u@N = \frac{\sum_u \frac{relevant@N \cap recommanded@N}{relevant@N}}{n} \quad (8)$$

where, n represents the number of users.

Precision@N: indicates the ratio of relevant recommended items existing in the Top-N set.

$$precision_u@N = \frac{\sum_u \frac{relevant@N \cap recommanded@N}{recommanded@N}}{n} \quad (9)$$

F1_score@N: indicates the balance between recall and precision.

$$f1_{score_u}@N = 2 * \frac{recall_u@N * precision_u@N}{recall_u@N + precision_u@N} \quad (10)$$

Accuracy@N: indicate the percentage of correct predictions by the model.

$$Accuracy_u@N = \frac{nbr\ of\ correct\ predictions}{nbr\ of\ total\ predictions} \quad (11)$$

5. RESULTS AND DISCUSSION

5.1 Evaluation of performance

This section evaluated our recommendation list's effectiveness for each approach. To do that, we have used the strategy of the cross-validation holdout, in which our data is randomly splited into 80% for the train set and 20% for the test set. Otherwise, we have used the Top-N sample strategy to rank the items for each user and reduce the time consumed to rank all items in the dataset. In our case, we have taken 80 items.

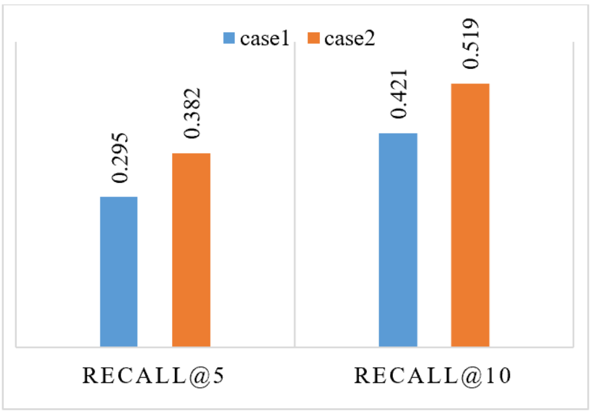

Figure 3. Recall measures for the two cases

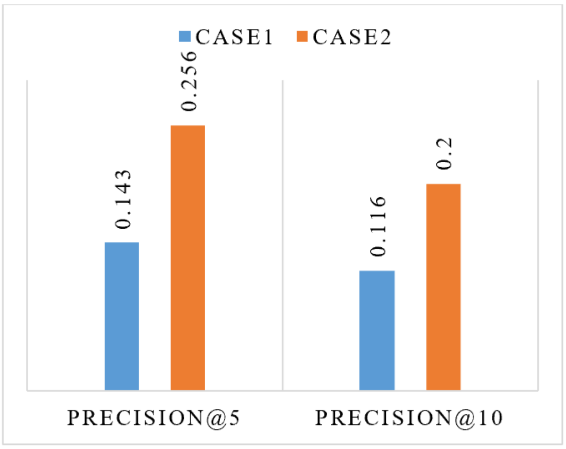

Figure 4. Precision measure for the two cases

To evaluate our system's quality performance, it is interesting to emphasize that the metrics used (accuracy, f1_score, precision, and recall) are modified according to the selected cases. In the first one, which is the case of a new student, we have adapted the cascading hybridization strategy between the SVD CF and the popularity-based. The recall measured 0.42 which means (42%) of the interacted articles of the test set are ranked by the model in top-10 items (Figure 3). It scored a precision of 0.116 (Figure 4.) which means (11.6%) of recommended resources are relevant in top-10 items. The f1_score is 0.495 (Figure 5.), and the accuracy is 0.856 (Figure 6.). Otherwise, in the second case, we have a standard student with information about his preferences in the database. We have provided the SVD-based collaborative filtering to generate the recommendation list. The recall measured 0.519 (Figure 3). In addition, the precision measured 0.20 (Figure 4), the f1_score measured 0.54 (Figure 5), and the accuracy measured 0.857 (Figure 6). All the results of the two cases are presented in the figures below.

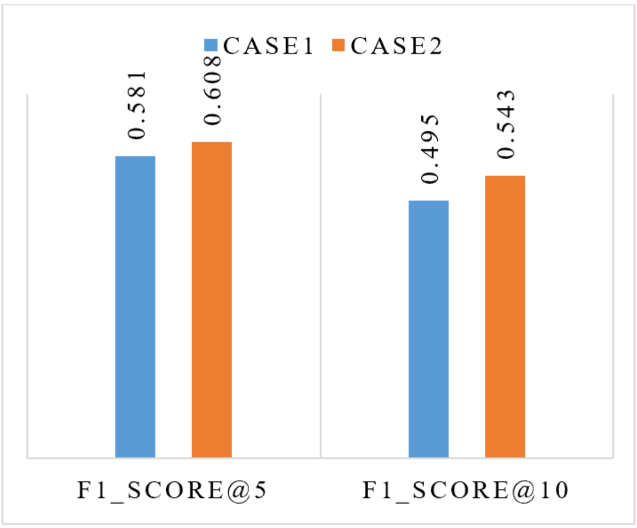

Figure 5. F1_score measure for the two cases

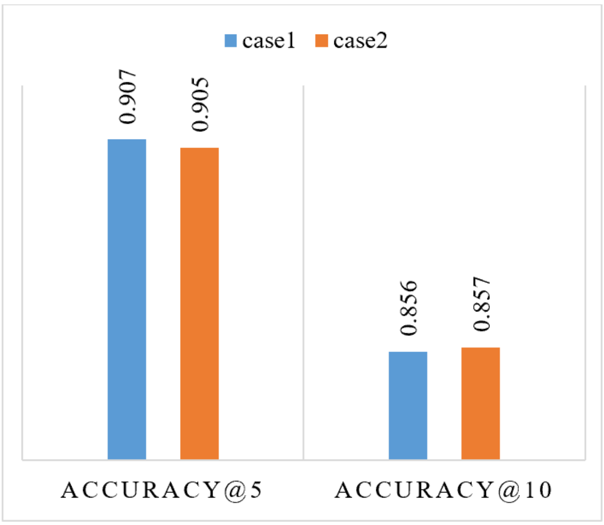

Figure 6. Accuracy measure for the two cases

Table 2. Users with a few interactions

	personId	interacted_count
1	-9212075797126931087	4
2	-9207251133131336884	2
4	-9196668942822132778	1
8	-9150583489352258206	1
11	-9099478998637725255	1
12	-9083704948999852989	2
14	-9060214117327732109	4
15	-9048557723087354030	1
18	-9012030317377670760	1
22	-8985529623369322698	1

To answer our questions about:

Does the proposed system provide a recommendation for new students and prevent user cold start problems?

We can take an example of our experiment with users with few (less than 5) interactions in the system (see Table 2).

When we consider the user with the ID "-9196668942822132778," who has only one interaction in the system, we can observe that our system provides a recommendation to him (Table 3). According to this example, we can conclude that our system that uses a hybrid approach can prevent the issue of cold-start and supply new learners with recommendations because our system adopts as baseline a popularity-based method to supply the most trending articles to new users.

According to our experiment, the system recorded encouraging results in the above figures. In addition, we have provided a LRMF technique enhanced with DNN architecture in the second case and compared it with SVD. The obtained results are presented in the Figure 7.

In Figure 7, the F1_score and the recall of SVD outperform those of the DNN. The precision of the DNN surpasses the SVD precision. However, the two models are very close in terms of accuracy.

5.2 Comparison and discussion

Many research studies have been conducted in the RS field in order to provide recommendations to users. A comparison between our proposed model and some other previous propositions is illustrated in Table 4. The recommender system presented in Mediani [24] proposed three models, including a content-based model using the TFIDF technique (CB_TFIDF), a collaborative filtering using SVD technique (CF_SVD), and a hybrid model combining CB_TFIDF and CB_TFIDF. While the work prsented in [29] uses a popularity approach, a collaborative filtering using SVD approach and a Hybrid model. This Table presents global recall and precision scores, ranking among the top 10 articles in the field of recommender systems for pedagogical resources.

Table 3. Some recommended resources for a given user with a few interactions

	interaction_weight	contentId	Title	Language
0	0.367803	-402970472570746	Former Google career coach shares a visual tri	Eng
1	0.367485	-862705118860535	When to make a Git Commit	Eng
2	0.367399	146958015103614	Don't document your code. Code your documentation.	eng
3	0.367213	569574447134368	Mastering Bash and Terminal	Eng
4	0.366964	-592047561263000	How it Feels to Learn Javascript in 2016	Eng
5	0.366812	237243848507014	The Continuous Delivery Maturity Model	Eng
6	0.366752	-751837351740148	My favorite people and resources to learn Andr...	Eng
7	0.366746	207244888783954	Welcome to GoogleBank, Facebook Bank, Amazon B...	Eng
8	0.366729	-665447003947831	Advanced Android Espresso	Eng
9	0.366726	255598321231014	The Definitive Guide to Database Version Control	Eng

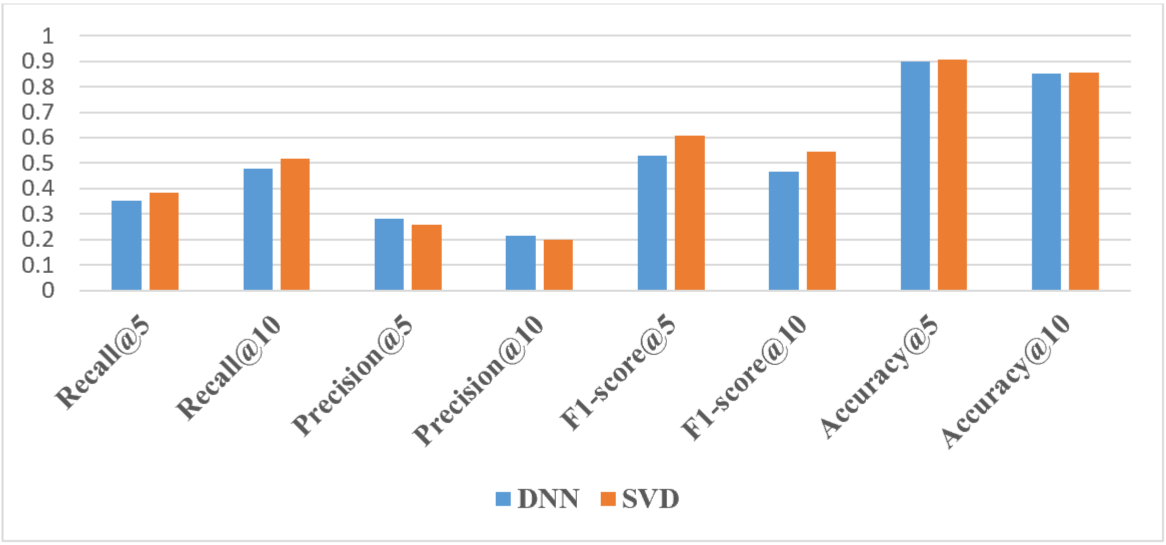

Figure 7. The difference between metrics of DNN and SVD in the case2

Table 4. Comparison of the methods of the recommender system

Ref.	Method	RECALL@10	PRECISION@10
	SVD	0.52	0.20
	DNN	0.50	0.22
[24]	CF_SVD	0.47	0.18
[24]	CB-TFIDF	0.26	0.12
[24]	HYBRID	0.48	0.19
[29]	POPULARITY	0.42	
[29]	SVD	0.46	
[29]	HYBRID	0.47	

As shown in the table above, our SVD and DNN methods recorded the best results according to the other results. We all used the same dataset. However, we have used some data engineering explained in the section above to better train the model. In addition, we have proposed a new architecture adapted to the E-learning platform allow us to tackle the cold start problem.

6. CONCLUSION

Recommendation systems have become an important tool for many websites and platforms. E-learning begins to integrate and ameliorate the recommender system provides learners with personalized and appropriate educational resources helping them obtain a learning experience of higher quality. This article proposed a collaborative filtering RS for teaching resources. Collaborative filtering usually encounters the problem of cold start, sparsty, and scalability. A new architecture is used for RS to eliminate the issue of cold start by switching between the cases of new and standard students. We have used the popularity and SVD-based CF in the first case by applying the cascading hybridization strategy to generate recommendations list for a new user. In the second case, where standard users are involved, we utilized the SVD-based collaborative filtering and LRMF enhanced with DNN architecture approaches to address issues of sparsity and scalability. This resulted in a list of relevant articles. The proposed model is verified on datasets containing articles. Accuracy, F1_score, precision, and recall are used to assess our system's performance. The results indicate that our system recommends a list of efficient and feasible recommendations to new users. Also, the SVD and DNN-based recommender system is crucial in recommending pedagogical resources to users concerning standard students.

We affirm that recommender systems enhanced by deep learning will act in future e-learning platforms. Our work could be ameliorated to achieve impressive results. As a perspective, we will use other approaches and techniques in our system to ameliorate the quality of the recommendations. In our proposed architecture, we will merge the content based methods enhanced by deep learning such as word embedding techniques to enable the RS to better comprehend words meaning and semantics in the content of educational resources.

REFERENCES

[1] Lu, J., Wu, D., Mao, M., Wang, W., Zhang, G. (2015). Recommender system application developments: A survey. Decision Support Systems, 74: 12-32. https://doi.org/10.1016/j.dss.2015.03.008

[2] Ansari, M. H., Moradi, M., NikRah, O., Kambakhsh, K. M. (2016). CodERS: A hybrid recommender system for an E-learning system. In 2016 2nd International Conference of Signal Processing and Intelligent Systems (ICSPIS), Tehran, Iran, pp. 1-5. https://doi.org/10.1109/ICSPIS.2016.7869884

[3] Zoppis, I., Manzoni, S., Mauri, G., Aragon, R.A.M., Marconi, L., Epifania, F. (2020). Attentional neural mechanisms for social recommendations in educational platforms. In Proceedings of the 12th International Conference on Computer Supported Education (CSEDU 2020), pp. 111-117. https://doi.org/10.5220/0009568901110117

[4] Kumar, R., Verma, B.K., Rastogi, S.S. (2014). Social popularity based SVD++ recommender system. International Journal of Computer Applications, 87(14): 33-37. https://doi.org/10.5120/15279-4033

[5] Bressan, M., Leucci, S., Panconesi, A., Raghavan, P., Terolli, E. (2016). The limits of popularity-based recommendations, and the role of social ties. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,

San Francisco, USA, pp. 745-754. https://doi.org/10.1145/2939672.2939797

[6] Shu, J., Shen, X., Liu, H., Yi, B., Zhang, Z. (2018). A content-based recommendation algorithm for learning resources. Multimedia Systems, 24(2): 163-173. https://doi.org/10.1007/s00530-017-0539-8

[7] Sabeima, M., Lamolle, M., Nanne, M.F. (2022). Towards Personalized Adaptive Learning in e-Learning Recommender Systems. International Journal of Advanced Computer Science and Applications, 13(8): 14-20. https://doi.org/10.14569/IJACSA.2022.0130803

[8] Ayub, M., Ghazanfar, M.A., Mehmood, Z., Saba, T., Alharbey, R., Munshi, A.M., Alrige, M.A. (2019). Modeling user rating preference behavior to improve the performance of the collaborative filtering based recommender systems. PloS One, 14(8): e0220129. https://doi.org/10.1371/journal.pone.0220129

[9] Jena, K.K., Bhoi, S.K., Malik, T.K., Sahoo, K.S., Jhanjhi, N.Z., Bhatia, S., Amsaad, F. (2022). E-Learning Course Recommender System Using Collaborative Filtering Models. Electronics, 12(1): 157. https://doi.org/10.3390/electronics12010157

[10] George, G., Lal, A.M. (2022). Hy-MOM: Hybrid recommender system framework using memory-based and model-based collaborative filtering framework. Cybernetics and Information Technologies, 22(1): 134-150. https://doi.org/10.2478/cait-2022-0009

[11] Lekshmi Priya, T., Sandhya, H. (2019). Matrix factorization for recommendation system. In International Conference on Artificial Intelligence and Data Engineering, Udupi, India, pp. 267-280. https://doi.org/10.1007/978-981-15-3514-7_22

[12] Burke, R. (2007). Hybrid web recommender systems. In The Adaptive Web. Lecture Notes in Computer Science, Berlin, 4321: 377-408. https://doi.org/10.1007/978-3-540-72079-9_12

[13] Burke, R. (2002). Hybrid recommender systems: Survey and experiments. User Model. User-Adapted Interact, 12(4): 331-370. https://doi.org/10.1023/A:1021240730564

[14] Symeonidis, P., Malakoudis, D. (2018). MoocRec.com: Massive Open online courses recommender system. Collaborative Recommendations, pp. 627-651. https://doi.org/10.1142/9789813275355_0019

[15] Rao, S., Salomatin, K., Polatkan, G., Joshi, M., Chaudhari, S., Tcheprasov, V., Gee, J., Kumar, D. (2019). Learning to be relevant: Evolution of a course recommendation system. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, pp. 2625-2633. https://doi.org/10.1145/3357384.3357817

[16] Natarajan, S., Vairavasundaram, S., Natarajan, S., Gandomi, A. H. (2020). Resolving data sparsity and cold start problem in collaborative filtering recommender system using linked open data. Expert Systems with Applications, 149: 113248. https://doi.org/10.1016/j.eswa.2020.113248

[17] Singh, M. (2020). Scalability and sparsity issues in recommender datasets: A survey. Knowledge and Information Systems, 62(1): 1-43. https://doi.org/10.1007/s10115-018-1254-2

[18] Venil. P, Vinodhini, G., Suban. R. (2019). A state of the art survey on cold start problem in a collaborative filtering system. International Journal of Scientific & Technology Research. 9(4): 2606-2612.

[19] Yuan, X., Han, L., Qian, S., Xu, G., Yan, H. (2019). Singular value decomposition based recommendation using imputed data. Knowledge-Based Systems, 163: 485-494. https://doi.org/10.1016/j.knosys.2018.09.011

[20] Ferreira, D., Silva, S., Abelha, A., Machado, J. (2020). Recommendation system using autoencoders. Applied Sciences, 10(16), 5510. https://doi.org/10.3390/app10165510

[21] Dadgar, M., Hamzeh, A. (2022). How to boost the performance of recommender systems by social trust? Studying the challenges and proposing a solution. IEEE Access, 10: 13768-13779. https://doi.org/10.1109/ACCESS.2022.3147955

[22] Alqallaf, S.S.E., Medhat, W.M., El-Shishtawy, T.A. (2022). A hybrid recommender framework for selecting a course reference books. Journal of Theoretical and Applied Information Technology, 100(4): 1004-1013.

[23] Mbaye, B. (2018). Recommender system: Collaborative filtering of e-learning resources. International Association for Development of the Information Society.

[24] Mediani, C. (2022). Interactive hybrid recommendation of pedagogical resources. Ingenierie des Systemes d'Information, 27(5): 695-704. https://doi.org/10.18280/isi.270502

[25] Roughgarden, T., Valiant, G. (2015). CS168: The modern algorithmic toolbox Lecture# 9: The singular value decomposition (SVD) and low-rank matrix approximations. http://theory.stanford.edu/~tim/s15/l/l9.pdf.

[26] Koren, Y., Bell, R., Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8): 30-37. https://doi.org/10.1109/MC.2009.263

[27] Tegene, A., Liu, Q., Gan, Y., Dai, T., Leka, H., Ayenew, M. (2023). Deep learning and embedding based latent factor model for collaborative recommender systems. Applied Sciences, 13(2): 726. https://doi.org/10.3390/app13020726

[28] Valcarce, D., Bellogín, A., Parapar, J., Castells, P. (2020). Assessing ranking metrics in top-N recommendation. Information Retrieval Journal, 23(4): 411-448. https://doi.org/10.1007/s10791-020-09377-x

[29] Mediani, Y., Gharzouli, M., Mediani, C. (2022). A hybrid recommender system for pedagogical resources. The International Conference on Digital Technologies and Applications, Morocco, pp. 361-371. https://doi.org/10.1007/978-3-031-02447-4_38