E-ISSN: 2676-6167 | P-ISSN: 2538-5100

Journal of Applied Research on Industrial Engineering

www.journal-aprie.com

J. Appl. Res. Ind. Eng. Vol. 10, No. 4 (2023) 615–636.

Paper Type: Research Paper

Leveraging Deep Learning Techniques on Collaborative Filtering Recommender Systems

Ali Fallahi Rahmat Abadi[1,*], Javad Mohammadzadeh[1]

[1] Department of Computer Engineering, Karaj Branch, Islamic Azad University, Karaj, Iran; ali.fallahi@kiau.ac.ir; j.mohammadzadeh@kiau.ac.ir.

Citation:

Fallahi Rahmat Abadi, A., & Mohamadzade, J. (2023). Leveraging deep learning techniques on collaborative filtering recommender systems. *Journal of applied research on industrial engineering, 10*(4), 615-636.

Received: 28/02/2021 Reviewed: 01/04/2021 Revised: 19/05/2021 Accepted: 21/06/2021

Abstract

With the exponentially increasing volume of online data, searching and finding required information have become an extensive and time-consuming task. Recommender systems as a subclass of information retrieval and decision support systems by providing personalized suggestions helping users access what they need more efficiently. Among the different techniques for building a recommender system, Collaborative Filtering (CF) is the most popular and widespread approach. However, cold start and data sparsity are the fundamental challenges ahead of implementing an effective CF-based recommender. Recent successful developments in enhancing and implementing Deep Learning architectures motivated many studies to propose Deep Learning-based solutions for solving the recommenders' weak points. In this research, unlike the past similar works about using Deep Learning architectures in recommender systems that covered different techniques generally, we specifically provide a comprehensive review of Deep Learning-based CF recommender systems. This in-depth filtering gives a clear overview of the level of popularity, gaps, and ignored areas on leveraging Deep Learning techniques to build CF-based systems as the most influential recommenders.

Keywords: Recommendation systems, Deep learning architectures, Collaborative filtering, Survey.

1 | Introduction

As a consequence of the exponential growth of online-based technologies, we have experienced many significant changes in various aspects of our life. In the digital age, individuals can store and access information in ways that were not accessible in the past. However, these brilliant opportunities caused some critical challenges. Searching for a specific record inside the continuously increasing amount of data becomes more complicated and confusing. As a subclass of decision support systems, recommender systems help users overcome the information overload and access what they need rapidly through personalized channels [1].

In the last decades, working with big data was a challengeable concern. However, today, extensive datasets are valuable resources for complex systems that work based on deep structured learning methods because of the recent development in artificial intelligence and computation power.

Corresponding Author: ali.fallahi@kiau.ac.ir

https://doi.org/10.22105/jarie.2021.275620.1264

Strength capabilities of Deep Learning-based approaches compared with traditional techniques for solving complicated problems caused significant attention to employing Deep Learning in various domains such as image processing [2], speech recognition [3], data mining [4], business [5], Natural Language Processing (NLP) [6], and information filtering systems like recommendation engines [7].

The recommender system's primary intention is to predict the user's tendency toward an item; the item can be a company's product, stock in a stock market, a friend in a social network [8], a movie, or a photo on a website, etc [9]. Based on this concept, many researchers proposed various recommenders with diverse functionalities to suggest books, films, music, hotels, friendships, ps, etc. [10], [11].

1.1 | Traditional Approaches to Build a Recommender System

Recommender systems can mainly be categorized into three types: Content-based, Collaborative Filtering (CF), and hybrid recommenders [12]. We illustrated this categorization in *Fig. 1*, which shows an overview of the three mentioned criteria. In the next following paragraphs, we will explain and pinpoint some notable aspects of each type of recommender system.

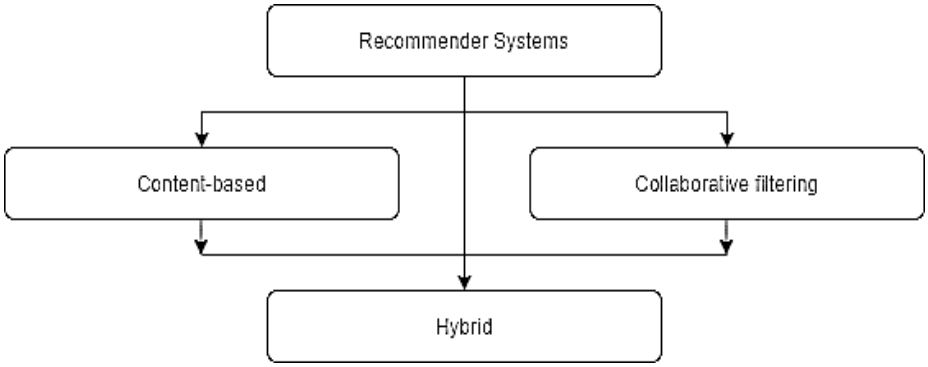

Fig. 1. Recommender systems categorization based on their techniques.

1.1.1 | Content-based

In this technique, the recommender engine considers the features of what users chose in the past to make suggestions for other related items of the dataset. Textual descriptions and tags are typical resources for implementing a content-based recommender [13], [14]. For instance, in a content-based book recommender system, if a user likes a book about Deep Learning, the system will first analyze the book's available textual properties, such as title, author, genres, complete text, etc. Then, various techniques like keyword-based vector-space structure will be used to find the other most similar books in the dataset with the highest similarity matching score with the chosen deep-learning book [15].

1.1.2 | Collaborative filtering

The fundamental hypothesis of CF is that users who had similar tastes and behaviors in the past will also have similar activities in the future. This strategy uses ratings or other measurable user's activities such as positive/negative comments, like/dislike, etc., to find similar users and then provide recommendations based on their similarities. Among all of the mentioned techniques, CF is the most well-known and popular method for implementing a recommender system [16], [17]. We describe the CF technique in detail in the following sections. There are two main approaches to build a CF system which are memory-based and model-based. The memory-based approach utilizes user rates to calculate the correlation among users or items. In a model-based approach, the primary step is to use the dataset to learn a proposed model. In the next step, the model is applied for making predictions. Matrix Factorization is the most common algorithm in building model-based CF systems [18].

1.1.3 | Hybrid recommenders

The critical factor in building an efficient recommender system is to improve accuracy and provide more personalized suggestions. Some studies tried to create hybrid systems based on a mixture of other techniques to benefit from the basic methods such as CF and content-based and overcome their drawbacks [19], [20]. Recent achievements in Deep Learning and neural networks also provided new opportunities for building hybrid systems that handle large amounts of data on complex networks [21], [22]. Hybridization can be done in different types. For instance, the system can provide recommendations based on various features generated by basic recommenders; this method is known as feature combination. Another approach is switching; the recommender system switches within different ways to provide recommendations [23].

1.2 | Related Studies

In the past few years, some studies such as Khan et al. [7], Da'u and Salim [9], Zhang et al. [10], Batmaz et al. [16] were conducted to review and survey Deep Learning-based recommender systems. However, to the best of our knowledge, not a single study specifically focused on leveraging Deep Learning techniques on CF recommenders as the most common technique to build recommender systems [24]. In the following lines, we introduce mentioned researches more by explaining their notable aspects.

Khan et al. [7] provided a comprehensive survey about Deep Learning-based rating prediction approaches. The authors reviewed different algorithms and architectures by concentrating on rating prediction systems; however, the main difference between the study and our research is providing an in-depth review by focusing on CF recommenders built based on Deep Learning architectures. By emphasizing presenting a Systematic Literature Review (SLR), Da'u and Salim [9] provided a survey about building a recommender system based on Deep Learning techniques. Zhang et al. [10] mainly focused on the taxonomical classification of reviewed studies and their approaches. Batmaz et al. [16], to help future researchers interested in the topic, categorized reviewed publications based on four dimensions: Deep Learning models and architecture, possible solutions for the challenges, recommender application domains, and purposive properties.

To provide a thorough study, we also reviewed acclaimed surveys about Deep Learning architectures, which were not limited to the subject of recommendation systems. For instance, Shrestha and Mahmood [25] proposed a review of Deep Learning algorithms and architectures by focusing on mathematical concepts of enhancing training operations. Although the study is not written on recommender systems, the authors flawlessly explained details about the structure of different Deep Learning architectures.

As a contribution, unlike past researches about using Deep Learning architectures in recommender systems that covered different techniques generally, in this study, we specifically provide a comprehensive review of Deep Learning-based CF recommender systems to guide and assist new researchers interested in the area.

The rest of the paper is organized as follows: Section 2 provides the preliminaries of recommender systems include traditional approaches and fundamental challenges. Deep Learning-based recommender systems are discussed in Section 3. Section 4 discusses the details of our results from some of the essential views applied to the topic. In Section 5, we present our conclusions and future work.

2 | Background

This section describes the CF technique with more details as the main focus of this study and the most commonly utilized method to build recommender systems. Moreover, in the following, we explain fundamental challenges ahead of implementing an efficient, accurate recommender.

2.1 | Collaborative Filtering Recommenders in Detail

In comparison with the content-based approaches, in CF, the system can provide recommendations based on the similarity of user's activities (or items' characteristics) without the necessity of analyzing the items [26]. *Fig. 2* shows a user-item rating matrix. This is a sample scenario of how a CF recommender engine predicts a specific user's rates for an item. In this example, ratings are between one to five. We selected Alice as the target user. The system aims to predict Alice's possible ratings for items that she did not rate. Items can be imagined as movies that she did not watch. Then recommend the items with the highest rates to her.

	Item1	Item2	Item3	Item4	Item5	Item6	Item7
Alice	4			5	1		
Bob	5	5	4				
Jim				2	4	5	
Kate		3					3

Fig. 2. A view of the user-item matrix.

The following paragraphs clarify the method step by step until proving the recommendation. The first step is calculating the similarity between Alice and the other three users. There are different metrics [27] for calculating similarity in recommender systems, such as:

2.1.1 | Jaccard similarity

In the below formula, the Jaccard Similarity of users p and q is calculated from the number of items rated both by the users p and q divided by the union of rated items by the users p and q. The fraction's numerator can be defined as the total number of the co-rated items between the users p and q. The denominator also can be defined as the total number of the rated items by both the users, p and q [28].

$$\text{JaccrdSim}_{p,q} = \frac{|r_p \cap r_q|}{|r_p \cup r_q|}. \quad (1)$$

Table 1 shows the result of calculating the Jaccard similarity between Alice and other users in the dataset.

Table 1. Result of the Jaccard similarity between Alice and other users.

	Bob	Jim	Kate
Alice	1/5	2/4	0

The above results indicate that Alice has the most similarity with Jim. However, the Jaccard similarity's main drawback is that the metric ignores how much two users are similar and only counts the number of co-rated items, whether the ratings are identical or the opposite [29].

2.1.2 | Cosine similarity

Cosine similarity measures similarity by calculating the cosine angle between the two rating vectors given by two targeted users. The smaller value of angle represents higher similarity and vice versa [30]. Cosine similarity is calculated as follows:

$$\text{Cos}_{p,q} = \frac{\overline{R_p} \cdot \overline{R_q}}{|\overline{R_p}| \cdot |\overline{R_q}|}. \quad (2)$$

In the Cosine similarity formula *Eq. (2)*, Rp and Rq respectively represent rating vectors of user p and q. In the numerator of the fraction, ".", indicates the dot product of two vectors.

To employ the Cosine similarity for the mentioned example, there should be some values for unrated items. The simplest way to complete this step is by adding zero to the empty cells. *Fig. 3* shows the user-item matrix after adding zero values to the empty cells to calculate the Cosine similarity between Alice and other users.

	Item1	Item2	Item3	Item4	Item5	Item6	Item7
Alice	4	0	0	5	1	0	0
Bob	5	5	4	0	0	0	0
Jim	0	0	0	2	4	5	0
Kate	0	3	0	0	0	0	3

Fig. 3. User-item matrix after adding zero values.

Table 2 shows the result of calculating the Cosine similarity between Alice and other users in the dataset.

Table 2. Result of the Cosine similarity between Alice and other users.

	Bob	Jim	Kate
Alice	0.93	0.75	0

In contrast to the Jaccard similarity, the results of the Cosine similarity indicate that Alice decides more similarly to Bob than Jim. By looking at the rating scores, it can be concluded Cosine similarity results are more realistic. However, treating missing ratings the same as the negative rates is a disadvantage of the Cosine similarity. In the example, we set all the empty cells with zero; in other words, we assigned an uncertain rate for unrated items, which can be utterly wrong. To clarify the problem, if, in a movie recommender, the system sets zero for unrated movies, the user may give it a high rating after watching it.

A solution to overcome this problem is to use the Centered Cosine. The Centered Cosine's main idea is to normalize ratings by subtracting each rate from the average of those ratings for the target user. Based on the explained situation, the concept in Centered Cosine similarity can be considered similar to the Pearson Correlation Coefficient (PCC).

2.1.3 | Pearson correlation coefficient

PCC [31] is one of the most widespread and notable popular similarity measure recommenders [32]. *Eq. (3)* shows the PCC formula.

$$PCC_{a,b} = \frac{\sum_{i=1}^{I_{a,b}} \left(r_{a,i} - \bar{r}_a\right)\left(r_{b,i} - \bar{r}_b\right)}{\sqrt{\sum_{i=1}^{I_a} \left(r_{a,i} - \bar{r}_a\right)^2} \sqrt{\sum_{i=1}^{I_b} \left(r_{b,i} - \bar{r}_b\right)^2}}. \quad (3)$$

In the PCC formula, *Eq. (3)*, $r_{a,i}$ indicates the rating score for item i, from the target user a. $r_{b,i}$ denotes the rating score for the same item from the user b. $\bar{r}_a$ and $\bar{r}_b$ mean the average rating of user *a*, and user *b* based on all rated items by each user.

	Item1	Item2	Item3	Item4	Item5	Item6	Item7
Alice	2/3			5/3	-7/3		
Bob	1/3	1/3	-2/3				
Jim				-5/3	1/3	4/3	
Kate		0					0

Fig. 4. The modified user-item matrix after subtracting the rates of each row from its average.

Fig. 4 shows the modified version of the user-item matrix after subtracting each rate from the average of ratings in that row. The final result of *Eq. (3)* is shown in *Table 3*.

Table 3. Result of the PCC similarity between Alice and other users.

	Bob	Jim	Kate
Alice	0.97	- 0.57	0

Typically, the second step of the CF is selecting Top-N most similar users to the target user. The concept is known as the K-Nearest Neighbors (KNN) method, and the main idea is to categorize other users' similarity values based on a predefined threshold [33]. Neighbors who, their score is greater than the threshold will be assigned to the Top-N group [34], [35]. The variable *N* can be set with different values. It had to be mention that all the members of this group must have rated the target item. In the above example, based on the PCC's result, it can be concluded that Bob is the most similar user to Alice in the dataset.

The final step is predicting the target user's rate for the target item. There are different formulas to do this step. However, to complete the example, we chose the average rating prediction model shown in *Eq. (4)* to provide predictions.

$$r_{x,i} = \frac{1}{k\sum_{y \in N^r Y^i}}. \quad (4)$$

In the equation *Eq. (4)*, $r_{x,i}$ is the predicted rate for item i from user x, which calculates the average rating of all the users in the selected neighborhood for item i. Also, the set N consists of users who rated item i and are similar to user x. Obviously, based on the total number of users in the presented example, Bob is the only similar user to Alice after selecting the most Top-N similar users. So, the predicted scores for unrated items by Alice will be similar to Bob's rates.

2.2 | Fundamental Challenges

Recommender systems are faced with different challenges. The increasing population of online users and numerous items caused difficulties such as sparsity, cold start, and the Grey sheep problem [36], [37]. This section outlines the mentioned challenges in making an efficient and accurate recommender system.

2.2.1 | Sparsity

Among the mentioned techniques for building a recommender, the CF method has a high dependency on the user's interaction with the system. However, in most datasets, there is a lack of sufficient data about users and items such as rates, comments, reviews, likes, dislikes, etc. Solving the sparsity problem was an appealing subject for many studies [38], [39]. While explicit trust relationships are used in many studies as a reliable approach to alleviating the data sparsity problem, some studies introduced propagation of trust and distrust values as a more practical solution to explore the unstated relationships

between users. As a result of these activities, a CF recommender has more data to calculate similarities and provide suggestions [40].

2.2.2 | Cold start

Cold start happens when a new user or a new item recently joined a system. There is not enough information about the user's activities in the past or enough interactions and feedback about the new item in this situation. The cold start has an extensive negative effect in CF-based recommender systems, since the recommender engine does not have enough information or feedback to calculate similarity [41]. Considering other origins of information such as social networks or contextual and demographic data is an efficient solution to overcome the cold start problem. In this regard, linked open data and DBpedia are two valuable resources to gain more information about users or items [42].

2.2.3 | Grey sheep

The problem of Grey Sheep users is a severe challenge of CF recommender systems. The "Gray Sheep" term refers to the users who are not similar to the majority of other users. This issue makes it difficult for recommender engines, especially the CF ones, to calculate the similarity between users and provide accurate suggestions [43], [44]. Researchers also tried to propose modern solutions to overcome the Grey Sheep problem with the development of machine learning techniques. For instance, using clustering algorithms is an effective solution to identify Grey Sheeps. Another approach is extracting content-based features from the Grey Sheep user's profile to improve recommendations' accuracy [45].

3 | Deep Learning-Based Recommender Systems

In recent years studying the influence of Deep Learning in different areas attained substantial interest. Likewise, in recommender systems, employing deep-learning techniques helped the experts enhance previous achievements and provide more accurate and precise results by prevailing over the fundamental challenges such as data sparsity, cold start, and Grey Sheep users [45], [46].

3.1 | Main Deep Learning Architectures in Collaborative Filtering Recommender Systems

In contrast to the past studies about applying Deep Learning architectures in recommender systems that made a general overview of different Deep Learning approaches, in this section, we expressly present a comprehensive analysis of Deep Learning-based CF recommender systems.

3.1.1 | Restricted boltzmann machines

Restricted Boltzmann Machine (RBM) is a special kind of Boltzmann machine. The RBM makes it possible to detect patterns in the input data by reconstructing them automatically. RBM is a network was built from two layers. Layers, respectively, are named visible and hidden layers. Each node in the first layer has a link with all the nodes in the hidden layer. The model is considered restricted because there is no connection between the nodes in the same layer [47]. *Fig. 5* shows an illustration of the RBM architecture.

Louppe [48] used parallel computing techniques with shared memory, distributed computing, and method ensembles for providing RBM-based CF systems. The author's experimental results indicate that parallel computing can be an effective solution to improve the provided suggestion's accuracy.

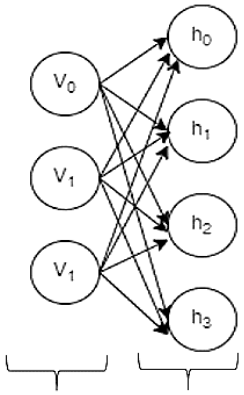

Fig. 5. A RBM architecture.

Georgiev and Nakov [49] introduced a joined user-based and item-based CF in a unified framework based on RBM. Moreover, the researchers employed real data in the first layer of the RBM architecture instead of multinominal variables. The authors also investigated the probability of mixing the RBM-based method's knowledge and the actual information.

Liu et al. [50] proposed a hybrid model based on RBM architecture and a CF approach. The authors used items' categories as the system's input to enhance the system performance and increase the result's accuracy.

Zheng et al. [51] introduced a CF Neural Autoregressive Distribution Estimation model named CF-NADE, which provides recommendations using RBM architecture. Authors showed that leveraging a Deep Learning network such as RBM can enhance the traditional and basic CF approaches.

Jia et al. [52] proposed a collaborative-based RBM recommender system for exhibition managements and participators in social events. The introduced recommendation framework mixes the data from various references and builds a relationship among the online knowledge and users who participated in the target event.

Du et al. [53] introduced an item-based RBM method for CF and applied the deep multilayer RBM network structure to overcome the sparsity problem. The authors considered every item as a separated RBM while each machine has similar properties such as weights and biases. The parameters are learned layer by layer in the deep network. They also used the batch gradient descent algorithm with minibatch to boost the convergence speed.

Wu et al. [54], to enhance the recommendations, considered trust relationships in recommender systems. The authors utilized explicit trust values and user's ratings as input data of the machine and proposed a social recommendation technique based on RBM.

3.1.2 | Autoencoders

Autoencoders are a neural network that takes an unlabeled set of inputs and reconstructs accurate results after encoding them. The system acts as a feature extraction engine and decides which data features are the most important. Autoencoders are generally a shallow network and consist of three layers: input, hidden, and output layers. A RBM can be considered as a two-layers Autoencoder. The system has two general steps, which are encoding and decoding. Typically, the features used to encode input for the hidden layer, also used for decoding and provide results in the output layer. The process of the forward propagation from the input layer toward the output layer and the backward propagation in a reverse

path is repeated continuously to achieve acceptable accuracy. *Fig. 6* shows an illustration of the RBM architecture [55].

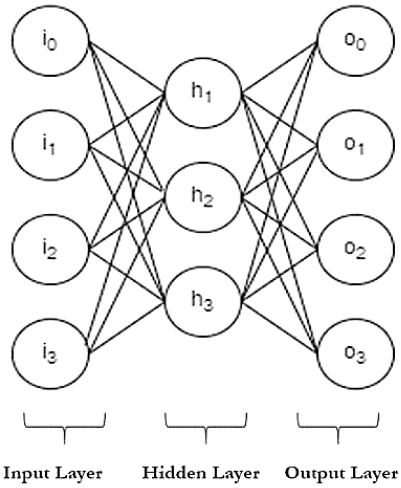

Fig. 6. An Autoencoder architecture.

Ouyang et al. [56] introduced one of the most pioneer studies in the autoencoder-based collaborative recommendation model. The authors utilized RBM and autoencoders together in the pretraining phase to achieve more personalized and accurate results. The system utilized a non-linear form of the input data.

Li et al. [57] proposed a recommender system, based on the mix of probability matrix factorization in CF with Autoencoder architecture. Due to their results, the model has significant efficiency in working with massive datasets such as movie or book recommenders.

Sedhain et al. [58] introduced the AutoRec recommender that uses user and item vectors based on Autoencoders architecture and CF approach.

Strub and Mary [59] proposed an autoencoders structure that calculates a non-linear matrix factorization based on the sparse ratings as inputs of the system.

Wang et al. [60] employed a stacked denoising autoencoders architecture to enhance rating prediction accuracy. The introduced model leverage content data and a CF approach based on user-item rating matrix values.

Wang et al. [61] implemented a Collaborative Recurrent Autoencoder (CRAE) that considers textual information and rating values. The authors technically used the hierarchical Bayesian model for denoising recurrent autoencoder.

Ying et al. [62] proposed a mixed model of a pair-wise recommender system that considers unexplicit feedback and CF concept. The authors employed Stacked Denoising Autoencoders (SDAE) to select the item description's characteristic features. The system utilized a Bayesian framework to combine rates and other data about items.

Wei et al. [63] proposed a recommender system based on CF and SDAE architecture to overcome the cold-start problem. The inputs of the model are the item's textual properties and user choices, and activities. An extended version of this research is proposed in [64].

Suzuki and Ozaki [65] employed users' ratings as inputs for an autoencoder architecture and computed the similarity between users in hidden layers. The decoded output of the system is a predicted rating used to provide recommendations.

Li and She [66] proposed a Bayesian generative multimedia recommender system based on a collaborative variational autoencoder. The system indicated both rating and content to explore the implicit connection between users and items.

Liang et al. [67] proposed a CF recommender for unexplicit feedback based on the Non-linear probabilistic. Technically the authors used Variational Autoencoders (VAEs) and a generative structure with multinomial probability and Bayesian reasoner for parameter calculation.

Li et al. [68] used different supplemental data such as item information, product tags, and shopping records to solve the data sparsity problem. The authors employed the autoencoder structure for every information source separately to have a better performance.

3.2 | Other Deep Learning Architectures in Recommender Systems

To make the study more comprehensive and provide a better understanding of the subject, in the following paragraphs, we present a brief explanation about the other Deep Learning architectures in recommender systems without limiting the references only to the CF approaches.

3.2.1 | Recurrent neural networks

Recurrent Neural Networks (RNN) are proper solutions for problems related to the changes in data patterns over time. The RNN has a feedback module that makes prediction possible for future input data. From the technical point of view, in a feed-forward neural network, signals flow in only one direction from input to output, one layer at a time. In RNN, the output of a layer is added to the next input and feedbacks to the same layer, which is typically the only layer in the entire network. The sequential pattern and the feature of changes in the hidden layer based on the information opened up RNN to various applications [69]. *Fig. 7* shows an illustration of the RNN architecture.

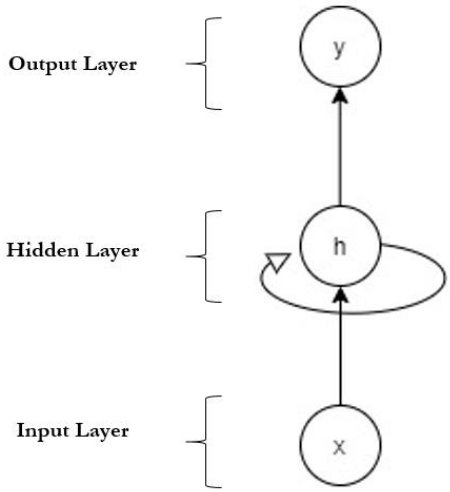

Fig. 7. A recurrent neural network architecture.

While there are studies about using RNN in different fields such as recommender systems, however, due to our comprehensive research, compared to the RBM and autoencoders, fewer studies employed the RNN to build a CF system. Ko et al. [70] proposed a collaborative RNN recommender system that uses a combination of contextual information with latent factors of user preferences to provide more accurate recommendations.

3.2.2 | Convolutional neural networks

Convolutional Neural Network (CNN) is one of the most dominant Deep Learning architectures in image processing and machine vision space [71], [72]. From a technical view, CNN is a kind of feed-forward neural network. However, in contrast to the typical neural networks, the convolution operation is used instead of ordinary matrix multiplication in CNN, the system convolves the input data, usually an array from pixels of an image, and analyzes the file, pixel by pixel, to detect edges and extract visual features. The depth of the network and matrixes dimensions vary due to the chosen architecture by the expert [73]. *Fig. 8* shows a sample of the CNN architecture.

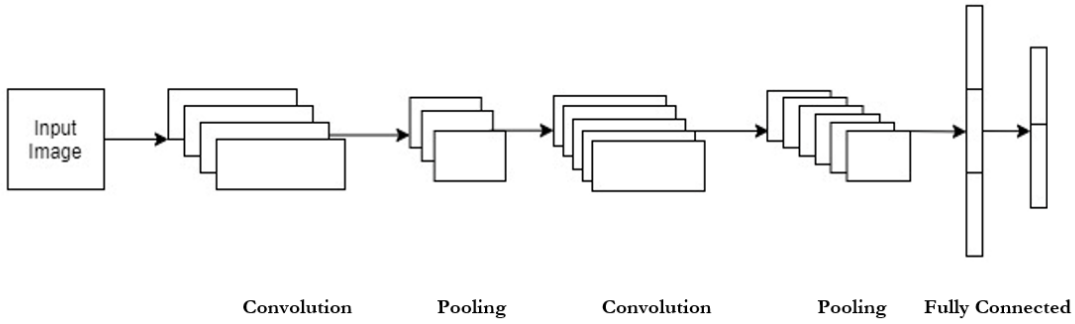

Fig. 8. A sample of the CNN architecture.

Zhang et al. [74], to solve the sparsity problem in CF recommenders, introduced Collaborative Knowledge base Embedding (CKE). The system works based on a hybrid CNN and Autoencoders architectures model to identify images' visual features. The authors also leveraged knowledge-based approaches to consider the dataset's content and textual information to provide a more robust user-item interaction space.

Low et al. [75] proposed a CNN CF recommender. The system uses the matrix factorization concept and creates connections between users and items.

Lee et al. [76] proposed a collaborative-based recommender engine that uses audio-visual features to calculate the similarity between videos. The system works based on the CNN architecture to extract visual points and can be highly profitable in video-sharing platforms.

He et al. [77] suggested utilizing a CNN to develop Neural Collaborative Filtering (NCF). The authors named their model ConvNCF. In the proposed structure to present the relation between users and items, the outer product was used alternatively of the dot product so that the system could catch the high-order similarities between defined aspects.

3.2.3 | Multilayer perceptron

Multilayer Perceptron (MLP) is one of the basic architectures of neural networks. The simple MPL consists of three layers which are input, hidden, and output layers. Each node in a MLP network is known as a perceptron. Obviously, the basic structure of MLP can not be considered as a deep neural network, while employing multiple hidden layers is a technique to develop the architecture and increase its potentials. The scheme of a MLP network is shown in Fig. 9. A MLP is an option to convert a linear technique of recommender system into a None-linear system. The MPL-based systems usually are used for supervised models. As a result of being a feed-forward network and having backpropagation, the system continuously modifies the weights and biases to improve the accuracy and achieve the expected result [78].

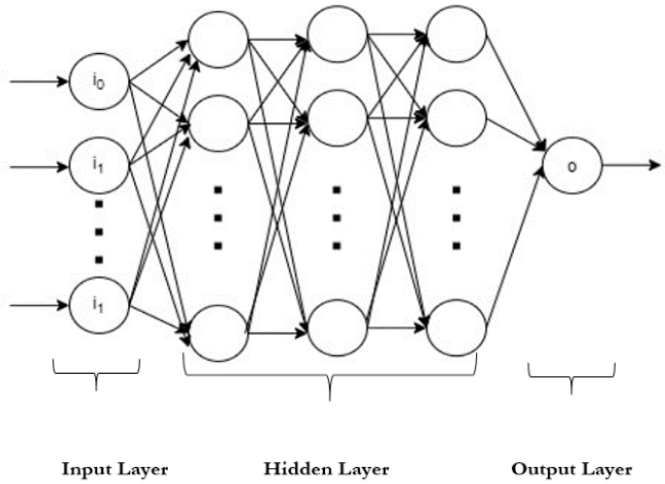

Fig. 9. A schema of a MLP architecture with multiple hidden layers.

Alizadeh [79] proposed a hybrid recommendation system based on the MLP network and CF concept. The authors addressed the cold start problem by leveraging the artificial neural network and content-based technique to utilize mutual information from users and items in the dataset. He et al. [80] used MLP as a neural network architecture to build a CF recommender system. The authors considered the user-item interaction function, and by leveraging Deep Learning advantages, improved the basic matrix factorization and CF and techniques.

3.2.4 | Deep belief networks

Combining RBMs together builds a more robust model which can solve the problems efficiently. The model is known as a Deep Belief Network (DBN). In a DBN, the hidden layer of each RBM is the visible layer of the next RBM; this relation continues in the same way for the next RBMs [81]. The last layer in a DBN can be used for clustering or classification. The conceptual architecture of a DBN network is shown in *Fig. 10*.

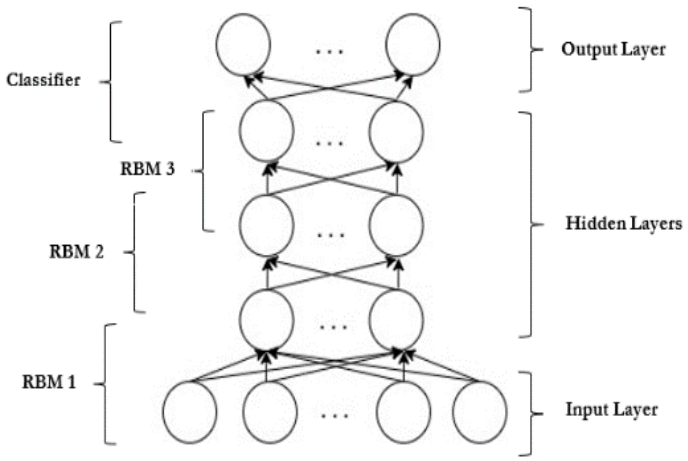

Fig. 10. A basic structure of DBN.

DBN generally combines both supervised and unsupervised learning steps. Compared with other Deep Learning architecture, DBNs need less labeled input data; this feature made DBNs a successful solution to implement real-world applications. Moreover, as DBN benefits from a deep network and multiple hidden layers, the system can provide more accurate results, especially compared to shallow nets [82]. Zhao et al. [83], to tackle the sparsity challenge in CF recommenders, introduced a hybrid system. The authors employed DBNs to discover user's characteristics. K-nearest neighbor technique is also used to select proper users and execute predictions.

3.2.5 | Attentional networks

The attention concept in computer science is formed based on the human ability to concentrate on a specific section of characteristics to perceive the target segments' value. With recent development in Deep Learning, attentional networks have become one of the popular topics in image processing, speech recognition, NLP, etc., and also recommender systems [84], [85].

Bahdanau et al. [86] proposed a model machine translation base on the sequence-to-sequence encoder-decoder approach. *Fig. 11* shows a graphical view of the proposed model.

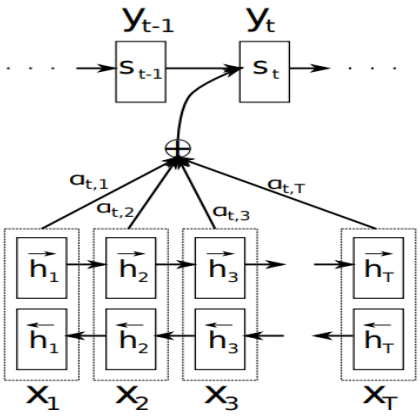

Fig. 11. Graphical view of proposed attention-aware model for translation [86].

In *Fig. 11*, the system's input is a sentence, and the output is its translation. To increase the result's similarity with human translation, the system tries to emphasize more on some specific input parts. X_1 to X_T and h_1 to h_T inside the rectangles show recurrence step activations. The $a_{t,1}$ to $a_{t,T}$ show how much attention is considered for the related input. Then the sum of the weights provides results boxes on top of the figure. S_t indicates the first time that generated y_1, etc.

Tay et al. [87] presented a memory-based Attentional networks architecture for collaborative metric learning. The authors introduced their model as Latent Relational Metric Learning (LRML). In the proposed system, user-item interactions were used as the information resource for the attention module. Jhamb et al. [88] implemented a contextual recommender system based on autoencoder neural network architecture and context-driven attention. The authors used the Attentional network to encode the contextual features into the hidden presentation of the user's characteristics.

3.2.6 | Generative adversarial networks

A Generative Adversarial Networks (GANs) typically consists of two main neural networks: generator and discriminator. The generator produces new samples of information, and the discriminator validates the generated data for its authenticity. GAN architecture became so popular in image processing Deep Learning-based systems [73], [89]. Tong et al. [90] proposed a Collaborative Generative Adversarial Network (CGAN) to build a recommender system. The authors used an auto-encoder as a generator module that takes features from user activities about items. Moreover, adversarial training was employed to enhance system efficiency and productivity.

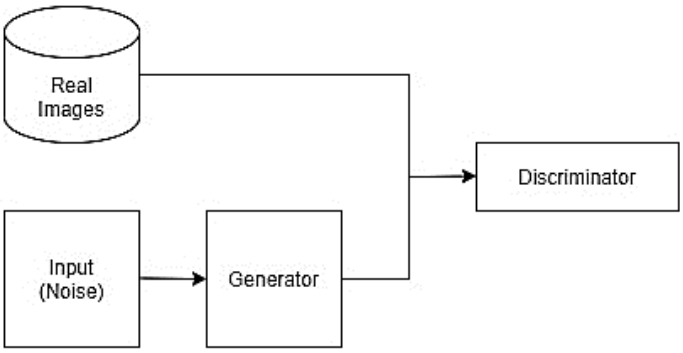

Fig. 12. A fundamental structure of the GAN.

Fig. 12 shows a GAN which produces fake images. The network has two inputs: a dataset of real images (top input) and a D-dimensional noise vector (bottom input). Generator Component produces fake images. In the next step, samples from real images and fake images become validate by the discriminator.

4 | Discussion

In this section, we discuss our work results and analyze the details from some of the most important views that can be applied to the topic. From *Table 4,* it is observed that Autoencoders and RBM are the most popular Deep Learning architecture for building CF-based recommender systems; and other approaches such as CNN, RNN, DBN, MLP, attentional networks and GAN have not received enough attention through the reviewed studies. Although, there could be different reasons for the popularity of Autoencoders and RBM, based on the provided data in *Table 5*, one of the critical differences is their training type which is unsupervised for both. Moreover, their potentials in common applications such as feature extraction and reducing dimensions are highly compatible with CF recommender systems structure.

Table 4. Literature on using Deep Learning architectures in CF recommender systems.

Deep Learning Architecture	Publication	Dataset
RBM	Louppe [48]	Netflix
	Georgiev and Nakov [49]	MovieLens
	Liu et al. [50]	MovieLens
	Zheng et al. [51]	MovieLens, Netflix
	Jia et al. [52]	Custom, Renren, Meetup
	Du et al. [53]	MovieLens
	Wu et al. [54]	MovieLens
	Ouyang et al. [56]	MovieLens
	Li et al. [57]	MovieLens, Book-Crossing, Advertising
	Sedhain et al. [58]	MovieLens, Netflix
	Strub and Mar. [59]	MovieLens, Jester
Autoencoders	Wang et al. [60]	Netflix, CiteULike
	Wang et al. [61]	Netflix, CiteULike
	Ying et al. [62]	CiteULike
	Wei et al. [63, 64]	Netflix
	Suzuki and Ozaki. [65]	MovieLens
	Li and She. [66]	CiteULike
	Liang et al. [67]	MovieLens, Netflix, Million Song
	Li et al. [68]	MovieLens, OfflinePay

Table 4. Continued.

Deep Learning Architecture	Publication	Dataset
RNN	Ko et al. [70]	Brightkite, LastFM
	Zhang et al. [74]	MovieLens, IntentBooks
CNN	Lo et al. [75]	MovieLens, Pinterest
	Lee et al. [76]	MovieLens, YouTube
	He et al. [77]	Yelp, Gowalla
MLP	Divan and Alizadeh [79]	MovieLens, Netflix
	He et al. [80]	MovieLens, Pinterest
DBN	Zhao et al. [83]	MovieLens
Attentional networks	Tay et al. [87]	MovieLens, Netflix, IMDb, LastFM, Books, Delicious, Meetup, Twitter
	Jhamb et al. [88]	MovieLens, Meetup
GANs	Tong et al. [90]	MovieLens, Netflix

Table 5 provides a condensed review and comparison of the different Deep Learning architectures. It has to be mention that some values, such as examples of the Common Applications presented in the table, also could be implemented in hybrid applications. Likewise, while RBMs are considered as generative models and "unsupervised" chose as their "training type", they can have components of the discriminative model and do their training phase in a supervised system [25].

Table 5. Deep Learning architectures comparison.

Deep Learning Architecture	Training Type	Training Algorithm	Common Applications
RBMs	Unsupervised	Gradient Descent	Feature Extraction
Autoencoders	Unsupervised	Backpropagation	Encoding; Reducing Dimensions
RNN	Supervised	Gradient Descent / Backpropagation	NLP; Translating Languages
CNN	Supervised	Gradient Descent / Backpropagation	Image Processing
MLP	Supervised	Gradient Descent / Backpropagation	Stochastic Solution; Fitness Approximation
DBNs	Supervised	Gradient Descent	Classification; Anomaly Detection
Attentional Networks	Supervised	Gradient Descent / Backpropagation	Image Processing; Speech Recognition, NLP
GANs	Unsupervised	Backpropagation	Generating Data; Reconstruction Data and Images

We also classified papers based on their datasets. *Table 6* shows datasets that researchers frequently used for building Deep Learning CF recommender systems. The values of this table demonstrate that MovieLens [91] is the most common dataset in building CF-based recommender systems that used Deep Learning architectures. MovieLens is a result of the GroupLens research project that the University of Minnesota did. It is a movie recommender website that users can rate the movies from 1, which means the worst score to 5, which means the maximum satisfaction. Two versions of the dataset were used in the reviewed studies, entitled MovieLens 100k and MovieLens 1M. Respectively, MovieLens 100k includes 100,000 ratings for 1,682 movies from 943 users, and MovieLens 1M includes 6,040 users who rated 1,000,000 ratings for 3,952 movies.

Table 6. Frequency of used datasets in the reviewed papers.

Title	Frequency
MovieLens	22
Netflix	10
CiteULike	4
Meetup	3
Pinterest	2
LastFM	2

Table 6. Continued.

Title	Frequency
YouTube	1
Twitter	1
IMDb	1
Jester	1
Book-crossing	1
Brightkite	1
Books	1
IntentBooks	1
GowaYelplla	1
Delicious	1
OfflinePay	1
Million Song	1
Renren	1
Advertising	1
Customized datasets	1

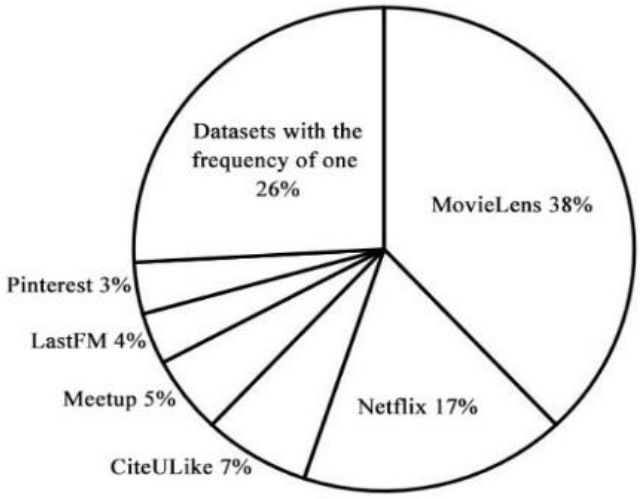

Fig. 13. Diversity of used datasets in the reviewed papers.

Fig. 13 shows a pie chart that presents another view of the given values in *Table 6*. However, to make a better presentation, datasets with the frequency of one, aggregated in one group, so the chart is divided into seven parts. Again, it is clear that the MovieLens dataset is the most common dataset in building Deep Learning CF recommenders. Another categorization in this work was done based on the publishing date of the reviewed papers; *Fig. 14* shows the diversity of these studies based on their publishing date.

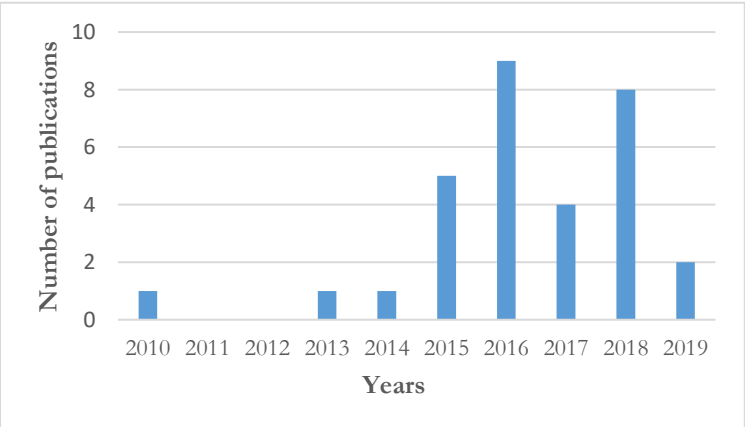

Fig. 14. Diversity of reviewed papers based on their publishing date.

As shown in *Fig. 14*, the year 2016 has the most significant number of published papers about CF-based recommender systems built based on deep leering architectures. Respectively, 2018 and 2015 are the second and third ones. As a consequence of these values, it can be concluded that there is a direct relationship between the studies about recommender systems and achievements in providing efficient Deep Learning architecture in recent years.

4.1 | Why Deep Learning Architectures for Recommendation?

With the tremendous extension in the volume of online data, handling the user's requests with traditional information retrieval and decision support systems has become highly challenging. However, as Deep Learning techniques are efficient in combining multiple sources of information and explore hidden features and patterns from them, big data caused increasing the popularity of these systems in recent years [16].

Another noticeable strength of employing deep neural networks, especially in CF-based recommender systems, is the capability in transforming the multi-dimensional user-item matrixes with sparsity into a smaller matrix with more data. For instance, Unger et al. [92] employed autoencoders to decrease the dimensions of environmental features and overcome the sparsity in context-aware recommendation systems.

Extracting visual features by CNN as complemental data to user's history and ratings is an adequate technique to solve the cold start problem [93]. Shin et al. [94] proposed a blog recommender system. The authors to deal with the cold-start problem combined derived features from textual data and pictures by CNN.

As mentioned in Section 2.2.3, identifying grey sheep users is another challenge ahead of building profitable recommender systems; due to the advantages of neural networks for clustering data based on different features. For instance, Rabba [95] proposed a system that detects grey sheep users based on unsupervised learning clustering techniques.

5 | Conclusion

Nowadays, practical filtering information and providing personalized recommendations have become increasingly critical, notably in online-based industries such as e-commerce or customer services. By increasing the interest in applying Deep Learning in different fields, enhancing recommender systems' performance by utilizing these kinds of approaches has also become increasingly pervasive.

In this study, we provided an extensive review of utilizing and leveraging Deep Learning architectures in CF recommender systems. However, in contrast to the subject's prior works that reviewed different techniques generally, we specifically provided a comprehensive review of Deep Learning-based CF recommender systems. We chose CF as the most common technique in building recommender systems and tried to clarify its relationship with Deep Learning architectures. Moreover, another analysis was done based on the researchs' datasets. Due to the results, the MovieLens dataset is the most popular dataset to make a CF recommender system that was build based on Deep Learning architectures.

Due to the results, Autoencoders and RBM are the most popular ones, and other architectures such as RNN, CNN, MLP, DBN, Attentional networks and GANs, gained fewer attention from researchers. In future work, we are interested in study the results of applying Deep Learning in varied fields of recommenders and compare the degree of influence from different perspectives, such as revenue, engagement, etc.

References

[1] Kolahkaj, M., Harounabadi, A., Nikravanshalmani, A., & Chinipardaz, R. (2021). Incorporating multidimensional information into dynamic recommendation process to cope with cold start and data sparsity problems. *Journal of ambient intelligence and humanized computing*, *12*, 9535–9554. https://link.springer.com/article/10.1007/s12652-020-02695-4

[2] Olague, G., Ibarra-Vázquez, G., Chan-Ley, M., Puente, C., Soubervielle-Montalvo, C., & Martinez, A. (2020). *A deep genetic programming based methodology for art media classification robust to adversarial perturbations* [presentation]. Advances in visual computing: 15th international symposium, ISVC 2020, (pp. 68–79). https://link.springer.com/chapter/10.1007/978-3-030-64556-4_6

[3] Dokuz, Y., & Tufekci, Z. (2021). Mini-batch sample selection strategies for deep learning based speech recognition. *Applied acoustics*, *171*, 107573. https://doi.org/10.1016/j.apacoust.2020.107573

[4] Keshtkar Langaroudi, M., & Yamaghani, M. (2019). Sports result prediction based on machine learning and computational intelligence approaches: A survey. *Journal of advances in computer engineering and technology*, *5*(1), 27–36.

[5] Panahandeh Khojin, G., Toloie Ashlaghi, A., & Afshar Kazmi, M. A. (2022). Provide an optimal model for determining and ranking inefficiency factors in the banking industry by combining data envelopment analysis and neural network. *Journal of decisions and operations research*, *7*(4), 610–627.

[6] Sadr, H., Pedram, M. M., & Teshnehlab, M. (2020). Multi-view deep network: a deep model based on learning features from heterogeneous neural networks for sentiment analysis. *IEEE access*, *8*, 86984–86997.

[7] Khan, Z. Y., Niu, Z., Sandiwarno, S., & Prince, R. (2021). Deep learning techniques for rating prediction: a survey of the state-of-the-art. *Artificial intelligence review*, *54*, 95–135.

[8] Mohamaddoust, R., Mohammadzadeh, J., Khalilian, M., & Nikravanshalmani, A. (2021). Measuring the community value in online social networks. *International journal of nonlinear analysis and applications*, *12*(Spec. Issue), 189–202.

[9] Da'u, A., & Salim, N. (2020). Recommendation system based on deep learning methods: a systematic review and new directions. *Artificial intelligence review*, *53*(4), 2709–2748.

[10] Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep Learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)*, *52*(1), 1–38.

[11] Alharthi, H., Inkpen, D., & Szpakowicz, S. (2018). A survey of book recommender systems. *Journal of intelligent information systems*, *51*, 139–160.

[12] Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, *17*(6), 734–749.

[13] Kolahkaj, M., Harounabadi, A., Nikravanshalmani, A., & Chinipardaz, R. (2020). A hybrid context-aware approach for e-tourism package recommendation based on asymmetric similarity measurement and sequential pattern mining. *Electronic commerce research and applications*, *42*, 100978. https://doi.org/10.1016/j.elerap.2020.100978

[14] Pradeep, N., Rao Mangalore, K. K., Rajpal, B., Prasad, N., & Shastri, R. (2020). Content based movie recommendation system. *International journal of research in industrial engineering*, *9*(4), 337–348.

[15] Cami, B. R., Hassanpour, H., & Mashayekhi, H. (2019). User preferences modeling using dirichlet process mixture model for a content-based recommender system. *Knowledge-based systems*, *163*, 644–655.

[16] Batmaz, Z., Yurekli, A., Bilge, A., & Kaleli, C. (2019). A review on deep learning for recommender systems: challenges and remedies. *Artificial intelligence review*, *52*, 1–37.

[17] Dakhel, A. M., Malazi, H. T., & Mahdavi, M. (2018). A social recommender system using item asymmetric correlation. *Applied intelligence*, *48*, 527–540.

[18] Zhang, R., Liu, Q., & Wei, J. X. (2014). *Collaborative filtering for recommender systems* [presentation]. 2014 second international conference on advanced cloud and big data (pp. 301–308). https://doi.org/10.1109/CBD.2014.47

[19] Riyahi, M., & Sohrabi, M. K. (2020). Providing effective recommendations in discussion groups using a new hybrid recommender system based on implicit ratings and semantic similarity. *Electronic commerce research and applications*, *40*, 100938. https://doi.org/10.1016/j.elerap.2020.100938

[20] Furtado, F., & Singh, A. (2020). Movie recommendation system using machine learning. *International journal of research in industrial engineering*, *9*(1), 84–98.

[21] Hajieskandar, A., Mohammadzadeh, J., Khalilian, M., & Najafi, A. (2020). Molecular cancer classification method on microarrays gene expression data using hybrid deep neural network and grey wolf algorithm. *Journal of ambient intelligence and humanized computing*, *14*, 5297–5307.

[22] Kalantari, S., Nazemi, E., & Masoumi, B. (2020). Emergence phenomena in self-organizing systems: a systematic literature review of concepts, researches, and future prospects. *Journal of organizational computing and electronic commerce*, *30*(3), 224–265.

[23] Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, *12*, 331–370.

[24] Gohari, F. S., & Tarokh, M. J. (2017). Classification and comparison of the hybrid collaborative filtering systems. *International journal of research in industrial engineering*, *6*(2), 129–148.

[25] Shrestha, A., & Mahmood, A. (2019). Review of deep learning algorithms and architectures. *IEEE access*, *7*, 53040–53065.

[26] Sachan, A., & Richariya, V. (2013). A survey on recommender systems based on collaborative filtering technique. *International journal of innovations in engineering and technology (IJIET)*, *2*(2), 8–14.

[27] Subha, V. S., & Dhanalakshmi, P. (2020). Some similarity measures of rough interval Pythagorean fuzzy sets. *Journal of fuzzy extension and applications*, *1*(4), 304–313.

[28] Lee, W. P., & Ma, C. Y. (2016). Enhancing collaborative recommendation performance by combining user preference and trust-distrust propagation in social networks. *Knowledge-based systems*, *106*, 125–134.

[29] Amirkhani, D., & Bastanfard, A. (2021). An objective method to evaluate exemplar-based inpainted images quality using Jaccard index. *Multimedia tools and applications*, *80*(17), 26199–26212.

[30] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). *Item-based collaborative filtering recommendation algorithms* [presentation]. Proceedings of the 10th international conference on world wide web (pp. 285–295). https://dl.acm.org/doi/abs/10.1145/371920.372071

[31] Pindyck, R. S., Rubinfeld, D. L., Pindyck, R. S., & Rubinfeld, D. L. (1998). *Econometric models and economic forecasts* (Vol. 4). Irwin/McGraw-Hill Boston.

[32] Sheugh, L., & Alizadeh, S. H. (2015). A note on pearson correlation coefficient as a metric of similarity in recommender system. *2015 AI & robotics (IRANOPEN)* (pp. 1-6). IEEE.

[33] Nourahmadi, M., & Sadeqi, H. (2021). A Clustering of Investors' Behavior According to Their Financial, Behavioral, and Demographic Characteristics (an Application of K-means Algorithm). *Innovation management and operational strategies*, *2*(2), 180–194.

[34] Chen, R., Hua, Q., Chang, Y. S., Wang, B., Zhang, L., & Kong, X. (2018). A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks. *IEEE access*, *6*, 64301–64320.

[35] Herlocker, J., Konstan, J. A., & Riedl, J. (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information retrieval*, *5*, 287–310.

[36] Alabdulrahman, R., & Viktor, H. (2021). Catering for unique tastes: Targeting grey-sheep users recommender systems through one-class machine learning. *Expert systems with applications*, *166*, 114061. https://doi.org/10.1016/j.eswa.2020.114061

[37] Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-based systems*, *46*, 109–132.

[38] Zhang, F., Qi, S., Liu, Q., Mao, M., & Zeng, A. (2020). Alleviating the data sparsity problem of recommender systems by clustering nodes in bipartite networks. *Expert systems with applications*, *149*, 113346. https://doi.org/10.1016/j.eswa.2020.113346

[39] Feng, C., Liang, J., Song, P., & Wang, Z. (2020). A fusion collaborative filtering method for sparse data in recommender systems. *Information sciences*, *521*, 365–379.

[40] RahmatAbadi, A. F., & Alizadeh, S. H. (2018). Effect of distrust propagation to enhance the performance of trust based recommender systems. *2018 9th conference on artificial intelligence and robotics and 2nd Asia-Pacific international symposium* (pp. 1-6). IEEE.

[41] Jakomin, M., Bosnić, Z., & Curk, T. (2020). Simultaneous incremental matrix factorization for streaming recommender systems. *Expert systems with applications*, *160*, 113685. https://doi.org/10.1016/j.eswa.2020.113685

[42] 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

[43] Zheng, Y., Agnani, M., & Singh, M. (2017). Identification of grey sheep users by histogram intersection in recommender systems. *Advanced data mining and applications: 13th international conference, ADMA 2017* (pp. 148-161). Springer International Publishing.

[44] Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., & Sartin, M. (1999). *Combing content-based and collaborative filters in an online newspaper* [presentation]. Proc. of workshop on recommender systems-implementation and evaluation. https://core.ac.uk/download/pdf/47187477.pdf

[45] Ghazanfar, M. A., & Prügel-Bennett, A. (2014). Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems. *Expert systems with applications*, *41*(7), 3261–3275.

[46] Feng, X., Zhang, H., Ren, Y., Shang, P., Zhu, Y., Liang, Y., … Xu, D. (2019). The deep learning--based recommender system "Pubmender" for choosing a biomedical publication venue: Development and validation study. *Journal of medical internet research*, *21*(5), e12957. https://www.jmir.org/2019/5/e12957/

[47] Salakhutdinov, R., Mnih, A., & Hinton, G. (2007). *Restricted boltzmann machines for collaborative filtering* [presentation]. Proceedings of the 24th international conference on machine learning (pp. 791–798). https://dl.acm.org/doi/abs/10.1145/1273496.1273596

[48] Louppe, G. (2010). *Collaborative filtering: Scalable approaches using restricted Boltzmann machines*. https://hdl.handle.net/2268/74400

[49] Georgiev, K., & Nakov, P. (2013). A non-iid framework for collaborative filtering with restricted boltzmann machines. *International conference on machine learning* (pp. 1148-1156). PMLR.

[50] Liu, X., Ouyang, Y., Rong, W., & Xiong, Z. (2015). Item category aware conditional restricted boltzmann machine based recommendation. *Neural Information processing: 22nd international conference, ICONIP 2015* (pp. 609-616). Springer International Publishing.

[51] Zheng, Y., Tang, B., Ding, W., & Zhou, H. (2016). A neural autoregressive approach to collaborative filtering. *International conference on machine learning* (pp. 764-773). PMLR.

[52] Jia, X., Li, X., Li, K., Gopalakrishnan, V., Xun, G., & Zhang, A. (2016). Collaborative restricted Boltzmann machine for social event recommendation. *2016 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM)* (pp. 402-405). IEEE.

[53] Du, Y., Yao, C., Huo, S., & Liu, J. (2017). A new item-based deep network structure using a restricted Boltzmann machine for collaborative filtering. *Frontiers of information technology & electronic engineering*, *18*(5), 658–666.

[54] Wu, X., Yuan, X., Duan, C., & Wu, J. (2019). A novel collaborative filtering algorithm of machine learning by integrating restricted Boltzmann machine and trust information. *Neural computing and applications*, *31*(9), 4685–4692.

[55] Zhang, G., Liu, Y., & Jin, X. (2020). A survey of autoencoder-based recommender systems. *Frontiers of computer science*, *14*, 430–450.

[56] Ouyang, Y., Liu, W., Rong, W., & Xiong, Z. (2014). Autoencoder-based collaborative filtering. *Neural Information Processing: 21st International Conference, ICONIP 2014* (pp. 284-291). Springer International Publishing.

[57] Li, S., Kawale, J., & Fu, Y. (2015). *Deep collaborative filtering via marginalized denoising auto-encoder* [presentation]. Proceedings of the 24th acm international on conference on information and knowledge management (pp. 811–820). https://dl.acm.org/doi/abs/10.1145/2806416.2806527

[58] Sedhain, S., Menon, A. K., Sanner, S., & Xie, L. (2015). *Autorec: autoencoders meet collaborative filtering* [presentation]. Proceedings of the 24th international conference on world wide web (pp. 111–112). https://dl.acm.org/doi/abs/10.1145/2740908.2742726

[59] Strub, F., Mary, J., & Philippe, P. (2015). *Collaborative filtering with stacked denoising autoencoders and sparse inputs* [presentation]. NIPS workshop on machine learning for ecommerce. https://inria.hal.science/hal-01256422/

[60] Wang, H., Wang, N., & Yeung, D. Y. (2015). *Collaborative deep learning for recommender systems* [presentation]. Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining (pp. 1235–1244). https://dl.acm.org/doi/abs/10.1145/2783258.2783273

[61] Wang, H., Shi, X., & Yeung, D. Y. (2016). Collaborative recurrent autoencoder: Recommend while learning to fill in the blanks. *Advances in neural information processing systems*, *29*. https://proceedings.neurips.cc/paper/2016/hash/0266e33d3f546cb5436a10798e657d97-Abstract.html

[62] Ying, H., Chen, L., Xiong, Y., & Wu, J. (2016). Collaborative deep ranking: A hybrid pair-wise recommendation algorithm with implicit feedback. *Pacific-asia conference on knowledge discovery and data mining* (pp. 555-567). Springer, Cham.

[63] Wei, J., He, J., Chen, K., Zhou, Y., & Tang, Z. (2016). Collaborative filtering and deep learning based hybrid recommendation for cold start problem. *2016 IEEE 14th international conference on dependable, autonomic and secure computing* (pp. 874-877). IEEE.

[64] Wei, J., He, J., Chen, K., Zhou, Y., & Tang, Z. (2017). Collaborative filtering and deep learning based recommendation system for cold start items. *Expert systems with applications*, *69*, 29–39.

[65] Suzuki, Y., & Ozaki, T. (2017). Stacked denoising autoencoder-based deep collaborative filtering using the change of similarity. *2017 31st International conference on advanced information networking and applications workshops (WAINA)* (pp. 498-502). IEEE.

[66] Li, X., & She, J. (2017). *Collaborative variational autoencoder for recommender systems* [presentation]. Proceedings of the 23rd acm sigkdd international conference on knowledge discovery and data mining (pp. 305–314). https://dl.acm.org/doi/abs/10.1145/3097983.3098077

[67] Liang, D., Krishnan, R. G., Hoffman, M. D., & Jebara, T. (2018). *Variational autoencoders for collaborative filtering* [presentation]. Proceedings of the 2018 world wide web conference (pp. 689–698). https://dl.acm.org/doi/abs/10.1145/3178876.3186150

[68] Li, T., Ma, Y., Xu, J., Stenger, B., Liu, C., & Hirate, Y. (2018). Deep heterogeneous autoencoders for collaborative filtering. *2018 IEEE international conference on data mining (ICDM)* (pp. 1164-1169). IEEE.

[69] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

[70] Ko, Y. J., Maystre, L., & Grossglauser, M. (2016). *Collaborative recurrent neural networks for dynamic recommender systems* [presentation]. Asian conference on machine learning (pp. 366–381). https://proceedings.mlr.press/v63/ko101.html

[71] Esfandiari, N., & Bastanfard, A. (2020). Improving accuracy of pedestrian detection using convolutional neural networks. *2020 6th Iranian conference on signal processing and intelligent systems (ICSPIS)* (pp. 1-6). IEEE.

[72] Sadr, H., Pedram, M. M., & Teshnehlab, M. (2021). Convolutional neural network equipped with attention mechanism and transfer learning for enhancing performance of sentiment analysis. *Journal of AI and data mining*, *9*(2), 141–151.

[73] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Convolutional networks. In *Deep learning* (Vol. 2016, pp. 330-372). Cambridge, MA, USA: MIT press.

[74] Zhang, F., Yuan, N. J., Lian, D., Xie, X., & Ma, W. Y. (2016). *Collaborative knowledge base embedding for recommender systems* [presentation]. Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 353–362). https://dl.acm.org/doi/abs/10.1145/2939672.2939673

[75] Low, Y. H., Yap, W. S., & Tee, Y. K. (2019). Convolutional neural network-based collaborative filtering for recommendation systems. *Robot intelligence technology and applications: 6th international conference, RiTA 2018* (pp. 117-131). Springer Singapore..

[76] Lee, J., Abu-El-Haija, S., Varadarajan, B., & Natsev, A. (2018). *Collaborative deep metric learning for video understanding* [presentation]. Proceedings of the 24th acm sigkdd international conference on knowledge discovery \& data mining (pp. 481–490). https://dl.acm.org/doi/abs/10.1145/3219819.3219856

[77] He, X., Du, X., Wang, X., Tian, F., Tang, J., & Chua, T. S. (2018). *Outer product-based neural collaborative filtering*. https://doi.org/10.48550/arXiv.1808.03912

[78] Rao Mangalore, K. K., Pradeep, N., Rajpal, B., Prasad, N., & Shastri, R. (2021). A survey on the techniques applied to the recognition and conversion of Indian sign language. *Journal of applied research on industrial engineering*, *8*(4), 399–411.

[79] Alizadeh, S. H. (2018). *A hybrid recommender system using multi layer perceptron neural network* [presentation]. 2018 8th conference of AI & robotics and 10th robocup Iranopen international symposium (Iranopen) (pp. 7–13). https://ieeexplore.ieee.org/abstract/document/8406624

[80] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T.-S. (2017). *Neural collaborative filtering* [presentation]. Proceedings of the 26th international conference on world wide web (pp. 173–182). https://doi.org/10.1145/3038912.3052569

[81] Hinton, G. E. (2009). Deep belief networks. *Scholarpedia*, *4*(5), 5947. http://dx.doi.org/10.4249/scholarpedia.5947

[82] Ouhbi, B., Frikh, B., Zemmouri, E., & Abbad, A. (2018). Deep learning based recommender systems. *2018 IEEE 5th international congress on information science and technology (CIST)* (pp. 161-166). IEEE.

[83] Zhao, C., Shi, J., Jiang, T., Zhao, J., & Chen, J. (2016). Application of deep belief nets for collaborative filtering. *2016 16th international symposium on communications and information technologies (ISCIT)* (pp. 201-205). IEEE.

[84] Xu, C., Feng, J., Zhao, P., Zhuang, F., Wang, D., Liu, Y., & Sheng, V. S. (2021). Long-and short-term self-attention network for sequential recommendation. *Neurocomputing*, *423*, 580–589.

[85] Zheng, L., Lu, C. T., He, L., Xie, S., He, H., Li, C., ... & Philip, S. Y. (2019). Mars: Memory attention-aware recommender system. *2019 IEEE international conference on data science and advanced analytics (DSAA)* (pp. 11-20). IEEE.

[86] Bahdanau, D., Cho, K., & Bengio, Y. (2014). *Neural machine translation by jointly learning to align and translate*. https://doi.org/10.48550/arXiv.1409.0473

[87] Tay, Y., Anh Tuan, L., & Hui, S. C. (2018). *Latent relational metric learning via memory-based attention for collaborative ranking* [presentation]. Proceedings of the 2018 world wide web conference (pp. 729–739). https://dl.acm.org/doi/abs/10.1145/3178876.3186154

[88] Jhamb, Y., Ebesu, T., & Fang, Y. (2018). *Attentive contextual denoising autoencoder for recommendation* [presentation]. Proceedings of the 2018 acm sigir international conference on theory of information retrieval (pp. 27–34). https://dl.acm.org/doi/abs/10.1145/3234944.3234956

[89] Li, Y., Wang, Q., Zhang, J., Hu, L., & Ouyang, W. (2021). The theoretical research of generative adversarial networks: an overview. *Neurocomputing*, *435*, 26–41.

[90] Tong, Y., Luo, Y., Zhang, Z., Sadiq, S., & Cui, P. (2019). Collaborative generative adversarial network for recommendation systems. *2019 IEEE 35th international conference on data engineering workshops (ICDEW)* (pp. 161-168). IEEE.

[91] Harper, F. M., & Konstan, J. A. (2015). The movielens datasets: History and context. *ACM transactions on interactive intelligent systems (TIIS)*, *5*(4), 1–19.

[92] Unger, M., Bar, A., Shapira, B., & Rokach, L. (2016). Towards latent context-aware recommendation systems. *Knowledge-based systems*, *104*, 165–178.

[93] Liu, J. Y. (2018). *A survey of deep learning approaches for recommendation systems* [presentation]. Journal of physics: conference series (Vol. 1087, p. 62022). 10.1088/1742-6596/1087/6/062022

[94] Shin, D., Cetintas, S., Lee, K. C., & Dhillon, I. S. (2015). *Tumblr blog recommendation with boosted inductive matrix completion* [presentation]. Proceedings of the 24th ACM international on conference on information and knowledge management (pp. 203–212). https://dl.acm.org/doi/abs/10.1145/2806416.2806578

[95] Alabdulrahman, R. (2020). *Towards personalized recommendation systems: domain-driven machine learning techniques and frameworks* (Ph.D Dissertation, Université d'Ottawa/University of Ottawa). https://ruor.uottawa.ca/handle/10393/41012