Neural Computing and Applications
https://doi.org/10.1007/s00521-020-05085-1

ORIGINAL ARTICLE

Social movie recommender system based on deep autoencoder network using Twitter data

Hossein Tahmasebi[1] · Reza Ravanmehr[1] · Rezvan Mohamadrezaei[1]

Received: 28 October 2019 / Accepted: 4 June 2020

Abstract
Recommender systems attempt to provide effective suggestions to each user based on their interests and behaviors. These recommendations usually match the personal user preferences and assist them in the decision-making process. With the ever-expanding growth of information on the web, online education systems, e-commerce, and, eventually, the emergence of social networks, the necessity of developing such systems is unavoidable. Collaborative filtering and content-based filtering are among the most important techniques used in recommender systems. Meanwhile, with the significant advances in deep learning in recent years, the use of this technology has been widely observed in recommender systems. In this study, a hybrid social recommender system utilizing a deep autoencoder network is introduced. The proposed approach employs collaborative and content-based filtering, as well as users' social influence. The social influence of each user is calculated based on his/her social characteristics and behaviors on Twitter. For the evaluation purpose, the required datasets have been collected from MovieTweetings and Open Movie Database. The evaluation results show that the accuracy and effectiveness of the proposed approach have been improved compared to the other state-of-the-art methods.

Keywords Social recommender system · Deep learning · Deep autoencoder network · Collaborative filtering · Content-based filtering · Social influence

1 Introduction

In recent years, a sudden increase in online information has caused confusion among users. Recommender systems are information filtering tools that direct users toward their interested, relevant products or services in a personalized way. Among various systems available to access information, the recommender system plays an important role in improving businesses and facilitating users' decision-making [1]. In general, the list of recommendations is created based on users' preferences and interests, items' properties, users' past interactions, and some additional information such as time and spatial data [2]. Recommender system models, based on the types of input data, are mainly classified into collaborative filtering (CF), content-based filtering (CB), and hybrid [3].

Recommender systems are evolving alongside the web; in recent years, they have been increasingly used in many areas of internet, such as e-commerce [4] and intelligent marketing [5], electronic book recommendation [6], music/movie recommendation [7], and tourism industry [8]. The objective of a recommender system is to make recommendations appropriate for each user, and therefore, these systems seek to balance accuracy, novelty, dispersion, and stability of the recommendations. Considering the ever-increasing amount of information in the virtual world, and the confusion of users in searching and selecting the required information, the use of recommender systems is increased.

One type of recommender system is social recommender systems [9]. The purpose of these systems is to reduce the information overhead by providing the most attractive and the most relevant content to the users of social media. In

✉ Reza Ravanmehr
r.ravanmehr@iauctb.ac.ir

Hossein Tahmasebi
hos.tahmasebi.eng@iauctb.ac.ir

Rezvan Mohamadrezaei
rez.mohamadrezaeilarki.eng@iauctb.ac.ir

[1] Department of Computer Engineering, Central Tehran Branch, Islamic Azad University, Tehran, Iran

Published online: 16 June 2020

fact, social media and recommender systems can enjoy mutual benefits. On the one hand, social media introduce various types of new data and metadata, such as votes and viewpoints, tags, and explicit online relations between users, which can be used by recommender systems in a unique way to increase efficiency. On the other hand, recommender systems are vital in social media websites to improve the acceptance and interaction with its users and thus play an important role in the overall success of social media. Is should be noted that some recommender systems, such as user-based collaborative filtering, are intrinsically social, because they are inspired by the natural process in which we humans seek advice or suggestions from others.

Deep learning models have recently shown great potential for guaranteed, effective learning and present the best performance in the areas of computer vision and natural language processing. Deep learning is a new area in machine learning researches, which is introduced to achieve one of the main goals of machine learning, i.e., artificial intelligence, and is being used widely for its superior capabilities in solving complex problems. Deep learning has recently led to outstanding advances in approaches related to recommender systems and has brought more opportunities in innovating recommendation methods to increase user satisfaction [10]. Recent advances in recommender systems based on deep learning have attracted considerable attention, by overcoming the barriers of conventional methods as well as achieving the highest quality recommendations. Deep learning can effectively achieve nonlinear and non-explicit user-item relations, furthermore, extracts complex relations between data from available data sources, such as background, textual, and visual information.

One of the first methods in collaborative filtering based on deep learning models has been presented by Ouyang et al. that utilized a deep autoencoder network [11]. After that, numerous works have been done on recommender systems using a deep autoencoder network. For example, the variation autoencoder networks have been used in [12–14], the denoising autoencoder networks have been employed in [15–18], the marginalized denoising autoencoders have been utilized in [19, 20], and the contractive autoencoders have been used in [21, 22].

The movie industry is one of the main areas of entertainment for people in the twenty-first century, and it is also one of the areas where researchers have extensively studied the recommender systems. With an enormous increase in the number of movies and considering the fact that humans inherently use the ideas of other people when making a choice, it appears that the social information related to movies and social information of users can be used to provide refined and personalized recommendations. This social information can be extracted from social networks such as Twitter and Facebook, employing simple techniques such as crawler development [23] or even complicated knowledge management systems based on semantic analysis and social information [24]. So far, various researches have been done on the movie recommender systems using valuable information from social networks [7, 25, 26]. Deep learning methods have also been discussed in various articles in this regard [27–29]. However, to the best of our knowledge, there have been very few articles that simultaneously use social networking information to calculate user social influence, as well as deep learning models to produce accurate recommendations.

In this article, a hybrid recommender system is developed, which generates accurate recommendations based on the user's social information and deep learning. This method employs a combination of collaborative and content-based filtering, taking into account the movie properties and the history of ratings given by users to the movies. To solve the cold start problem, the properties of movies seen by the user will be utilized using content-based filtering. Then, the ratings will be predicted for movies not seen by the user based on collaborative filtering and employing a deep autoencoder network. In this network, input data are reconstructed in the output layer, and the bottleneck layer is used to display the features of the input data.

In order to train the deep autoencoder network, the ratings and interests of the user with more social influence will be more effective. The social influence of each user will be calculated based on his/her activities on Twitter and a combination of the average published Tweet, the average number of retweets, and the average number of likes for the published tweets. Finally, those movies with the highest ratings will be recommended to the user. The results obtained from the evaluation and comparison of the proposed approach with the other state-of-the-art methods indicate the effectiveness of the proposed method in increasing the accuracy of the recommendations.

One of the major achievements of the proposed approach is the ability to use it for other recommender systems such as product recommender systems or e-learning. For this purpose, the history of users' ranking with more social influence can be utilized. Then, according to the proposed workflow in this research, the obtained feature sets can be used for training the deep autoencoder network, which can eventually lead to accurate and effective recommendations.

In brief, we summarize the contributions of this research as follows:

- Developing a hybrid approach for movie recommender system based on collaborative filtering, content-based

filtering, and deep autoencoder network, which is easily extendable to other areas of product-based recommender systems.
- Using the social influence of users, which is calculated based on their social activities on Twitter.
- Providing a new hybrid dataset from MovieTweetings and Open Movie Database (OMDB), which reduces sparsity and increases the information required to achieve higher accuracy.

In the remainder of this article, in section two, the previous literature is discussed. In the third section, the proposed method, which is a hybrid recommender system based on users' social information and deep autoencoder network, is presented. In Sect. 4, the evaluation and comparison of the proposed approach with the state-of-the-art methods are discussed. Finally, conclusion and future works are presented.

2 Previous literature

In this section, we provide the basic terminology and concepts regarding the deep autoencoder network and the recommender systems employing deep learning models.

According to the proposed approach utilizing a deep learning model based on information from social networks, we briefly review researches in this field. For this purpose, previous studies have been classified into two categories: movie recommender systems based on social information and based on deep learning models.

2.1 Deep autoencoder network

Deep learning has emerged as a new area of machine learning in recent years. Over the past few years, the techniques developed based on deep learning research have already been impacting a wide range of signal and information processing tasks. The deep learning process is based on the presentation and abstraction of data at several different levels [10]. In general, deep learning models include a hierarchical architecture with a large number of layers (deep architecture), and each layer contains a nonlinear information processing unit. These layers are used to extract and transfer the features for pattern analysis and classification through a supervised or unsupervised learning approach.

An autoencoder is an unsupervised deep learning method that is used to optimally encode datasets for dimension reduction [30]. The input data of the autoencoder are initially converted to abstract representation and then returned to the original format using the encoder function. In other words, it is trained to transform the input into a representation that can be reconstructed again. In fact, autoencoder attempts to approximate the identity function in this process. One of the key advantages of the autoencoder is the ability to extract useful features continuously during propagation and filter out the useless information. Furthermore, since the input vector transforms into a lower-dimensional representation in the coding process, the efficiency of the learning process can be enhanced [31, 32]. And last but not least, the autoencoder can be used in recommender systems for feature representations learned from user-item content features [10].

Autoencoder is a feed-forward neural network similar to a multilayer perceptron. The difference between a multilayer perceptron and autoencoder is that autoencoder reconstructs the input, and the aim of a multilayer perceptron is to predict target values with specific inputs. The number of available nodes in the input and output layers is the same. In the encoding process, autoencoder first converts the x vector into a hidden representation h, using weight matrix ω. Then, in the decoding process, the autoencoder converts h to its original format, so that $\sim x$ can be obtained with the weight matrix $\acute{\omega}$. Theoretically, $\acute{\omega}$ must be a transpose of ω. The parameter optimization is used to minimize the average reconstruction error between x and $\sim x$. Mean square errors (MSEs) are used to measure the accuracy of the reconstruction according to the expected distribution of input features.

Autoencoding is a function which mainly consists of two parts:

(a) *Encoder* a function to extract features, which calculates features vector from inputs [33]. Therefore, if we show features vector with h, encoder function with f_θ and encoding datasets with $x^{(t)}$, Eq. (1) is obtained:

$$h^{(t)} = f_\theta\left(x^{(t)}\right), \quad x^{(t)} = \left\{x^{(1)}, \ldots, x^{(T)}\right\} \tag{1}$$

where $h^{(t)}$ is features vector of $x^{(t)}$.

(b) *Decoder* a function g_θ which maps the feature space to the input space using Eq. (2):

$$r = g_\theta(h) \tag{2}$$

In fact, from a given probability function, probabilistic models are defined and trained to maximize (often approximately) data similarity. Autoencoders are trained using different training principles. The parameters set θ of both encoder and decoder is similarly trained to reconstruct the main input. It attempts to minimize the reconstruction error as much as possible. $L(x, r)$ is reconstruction error that is obtained from Eq. (3). It is a measure of the discrepancy between x and its reconstruction r [33]:

$$L(x, r) = \|x - r\|^2 \tag{3}$$

In summary, autoencoder training involves finding the parameter vector θ to minimize the reconstruction error using Eq. (4) where $x^{(t)}$ is a training example:

$$\mathcal{J}_{AE}(\theta) = \sum_{t} L\left(x^{(t)}, g_\theta\left(f_\theta\left(x^{(t)}\right)\right)\right) \quad (4)$$

Minimizing this value is usually performed by the stochastic gradient descent method similar to the multilayer perceptron training procedure [33].

2.2 Recommender systems based on deep learning

Various machine learning methods have always been used in different applications of recommender systems based on social information. Crespo et al. proposed an intelligent personalized recommendation system for electronic books using data gathered from user interaction [6]. The results of the proposed system showed that it could provide useful personalized recommendations to the users and solved the problem of information overload. Núñez et al. performed sentiment analysis on Twitter for tourism-based recommender applications [8]. For this purpose, the authors employed natural language processing (NLP) techniques, together with computational linguistics, to identify and extract subjective information from Twitter.

Recently, various companies and institutes use deep learning to increase the quality of their recommendations. For example, Covington et al. presented a new algorithm for YouTube video recommendations based on deep neural networks [34]. Cheng et al. provided a recommender system application for Google Play employing a deep model [35]. Okura et al. introduced an RNN-based news recommender system for Yahoo [36]. All these models have been evaluated by many users, and the results have shown significant improvement over traditional models.

Different methods of deploying deep learning models in recommender systems are classified into two general categories: neural network and deep hybrid models. The neural network models include multilayer perceptron (MLP), autoencoder (AE), convolutional neural network (CNN), recurrent neural network (RNN), restricted Boltzmann machine (RBM), neural autoregressive distribution estimation (NADEs), attentional model (AM), adversarial network (AN) and deep reinforcement learning (DRL) [10]. The hybrid deep learning models are composed of two or more deep learning techniques. The flexibility of deep neural networks makes it possible to combine several neural blocks to create a more robust hybrid model.

It should be noted that the proposed recommender system in this research falls into the category that has only utilized one of the deep learning techniques, the deep autoencoder network.

2.3 Movie recommender systems based on social information

A hybrid recommender algorithm has been developed by Li et al. that calculates the similarity between users based on the integration of movie features and the interest of users [7]. The authors combined the features vector of movies and the matrix of user ratings together and then created and iteratively updated the vector of user interests. Then, the similarities between users have been calculated based on this vector of integrated interests.

The aim of Li et al. was to provide a two-step approach to tackle the problem of cold start in recommender systems and improve the quality of recommendations [26]. The semantic relations between items are examined based on textual information and are extracted using lower-level matrix analysis. These semantic relations increase the capacity of semantic analysis and combine the ranking of users' ratings by decomposing a higher level matrix in the learning model.

Li et al. presented a recommender system that extracted the priority of information and preferences of users from microblogs to assess the similarity between online movies and TV series [37]. The first attempt is to eliminate the gap between movie and TV series' viewers, which is done by overviewing the activities of users on social networks. Then, similar programs of social networks are used to suggest programs in other media devices.

Sun et al. proposed an idea that shows in social recommender systems, users' relationships in social networks can improve the accuracy of recommendations [38]. They proposed a matrix factorization framework with social rules. For this purpose, a biclustering algorithm has been used to identify the most suitable group of friends for generating various final recommendations.

A personalized recommender system has been presented by Seo et al. based on reinforcing of friendship between users, which recommends interesting topics to users [39]. In this paper, the metrics of reinforcement are introduced as an alternative to the similarity measurements in traditional social recommender systems. Furthermore, in this article, Twitter is used to analyze large social data.

Zhao et al. focused on solving data sparsity problem through training a multimodal network representation for ranking movie recommendations [40]. In this paper, a heterogeneous social-movie recommender system called MNRL (multimodal network representation learning) is presented, which benefits from textual descriptions and posters of movies alongside social relationships and ratings

of users. The movie posters are fed into a convolutional neural network as visual symbols.

Pérez-Marcos et al. provided a hybrid recommender system for video games that obtains implicit ratings from the hours of play of the users [41]. The authors utilized both collaborative and content-based filtering using data obtained from video game platforms like steam that includes users' account properties, achievements, played hours, reviews, etc.

Katarya and Verma proposed a movie-based recommender system employing the bio-inspired gray wolf optimizer algorithm and fuzzy *c*-mean (FCM) clustering technique [42]. The proposed system predicts the rating of a movie for a particular user based on his historical data and the similarity of users.

A social movie recommendation approach using implicit social relations of users has been provided by Ling et al. [43]. To mitigate the inherent drawbacks of explicit social relations, they proposed a social recommendation method using implicit friends extraction from heterogeneous information network (IFHN). IFHN obtained the rating prediction using an extended probabilistic matrix factorization model.

Table 1 summarizes the existing works discussed in this subsection and addresses the techniques, advantages/disadvantages, and dataset of each approach.

2.4 Movie recommender systems based on deep learning

Lee et al. attempted to combine collaborative filtering and deep learning network and presented mDA-CF and mSDA-CF approaches by incorporating the probabilistic matrix factorization and marginalized/stacked denoising autoencoders (mDA/SDA) [20]. Denoising autoencoders reconstruct the input from a corrupted version of the data aimed at learning a more robust mapping from the data.

A movie recommender system based on deep learning has been presented by Behera et al. that benefits from user ratings to movies [27]. A model is developed employing a restricted Boltzmann machine (RBM) to predict the missing entries in the dataset. Their method consists of two parts: item-based and user-based collaborative filtering.

Deldjoo et al. demonstrated users' interests based on the visual aspects of the film that reflect the design and genre of the film (colors and backgrounds) [28]. In the proposed method, the MPEG-7 visual descriptor and hidden layers of deep learning are used, which automatically calculate the feature of each video file.

Wei et al. attempted to solve both complete/incomplete cold start problems using IRCD-CCS and IRCD-ICS methods, respectively [29]. For this purpose, the authors developed a specific deep neural network based on self-adaptive differential evolution (SADE), which is used to extract the content features of the items. Then, timeSVD++, which models temporal dynamics of user preferences and item features, is customized to take the content features into the prediction of ratings for cold start items.

A-COFILS (autoencoder-collaborative filtering to supervised learning) has been proposed by Barbieri et al. to transform a collaborative filtering problem to classical supervised learning [44]. The authors experimented with stacked denoising autoencoder (SDA) instead of the singular value decomposition (SVD) method to learn more useful and complex representations in the proposed neural network.

An autoencoder has been utilized by Strub et al. in the process of producing recommendations [45]. They created a relation between matrix factorization method and deep neural networks. Autoencoder is used to estimate incomplete data and alleviate the cold start problem.

Sedhain et al. employed a deep autoencoder network to find unknown user ratings [46]. This method is a combination of a collaborative filtering approach and an autoencoder network that is capable of learning nonlinear data representations.

A hybrid recommender system using a deep neural network has been proposed by Kiran et al. [47] that combines the deep learning features with side information available about users and items. Each hidden layer computes a linear function followed by a LeakyReLU activation function. The ReLU (rectified linear unit) activation function is one of the most popular nonlinear activation functions used in neural networks.

Gai et al. proposed a deep transfer collaborative filtering (DTCF) for movie recommendation system based on the combination of collaborative filtering and deep learning approaches [48]. For this purpose, DTCF employed both nonnegative matrix tri-factorization and stacked denoising autoencoder to improve recommendation performance.

Table 2 summarizes the existing works discussed in this subsection and addresses the techniques, advantages/disadvantages, and dataset of each approach.

3 Proposed method

As mentioned in the previous sections, the purpose of this paper is to introduce a hybrid recommender system that provides accurate and effective recommendations using social data, users' preferences and interests, and movie features. The proposed method employs a deep autoencoder network, which reduces the problem of data sparsity. Figure 1 shows a brief overview of the proposed approach,

Table 1 Overview of social movie recommender systems approaches

References	Techniques used	Pros(⊕)/Cons(–)	Dataset
[7]	*K*-nearest neighbor algorithm Collaborative filtering	⊕ Good performance in the case of data sparsity ⊕ Tracking the change of user interests ⊕ Bridging the movie feature and user interest – Considering only MAE for evaluation	MovieLens 100K
[26]	Weighted textual matrix factorization Two-level matrix factorization	⊕ Considering textual information of items to solve sparsity and cold-start ⊕ Novel matrix factorization – Not using other deeper semantic computation methods such as ontologies	MovieLens 1M, BookCrossing
[37]	Collaborative filtering Microblog Recommendation Knowledge discovery Behavior analysis Association rule	⊕ Sentiment analysis approach ⊕ Presenting personalized suggestion ⊕ Solving the cold-start problem – Only use collaborative filtering	Netflix Prize
[38]	Matrix factorization Social regularization Biclustering algorithm	⊕ Considering friendship among users – Cold-start problem – Ignoring social influence	Del.icio.us 2013
[39]	Collaborative filtering Friendship strength	⊕ Introducing a novel measure to calculate the closeness between users in a social circle ⊕ Presenting personalized suggestion – Only use collaborative filtering	Twitter, Daumsoft
[40]	Recurrent neural network Convolutional neural network Random-walk learning with multimodal heterogeneous neural networks	⊕ Utilizing RNN and CNN together to represent movies in terms of their textual synopsis and poster images ⊕ Solving the sparsity problem – Ignoring the spatiotemporal factors	Douban
[41]	Collaborative filtering Content-based filtering	⊕ Considering the number of hours played per video game – Ignoring social information	Steam Video Games
[42]	Collaborative filtering Gray wolf optimizer Fuzzy c-mean	⊕ Adopting a bio-inspired meta-heuristic algorithm –Ignoring demographics features, sentiments analysis, machine learning, and big data environments	MovieLens 100K
[43]	Matrix factorization	⊕ Incorporating implicit social information ⊕ Alleviating the cold-start problem – Ignoring user interest changes	FilmTrust, Douban

SRDNet (Social Recommender Deep autoencoder Network).

As shown in the above figure, first of all, we acquire MovieTweetings dataset and then add the details of movies and users' social information from OMDB and Twitter, respectively. SRDNet consists of two separate recommender engines: collaborative filtering and content-based filtering. Each filtering algorithm works separately and generates a list of recommendations. The collaborative filtering method used in this paper is user-based CF. It finds the similarities between users and recommends movies to users of the same interests also uses the concept of social influence. The content-based filtering is item based, which recommends movies to users by analyzing movie features. By using these two recommender engines, it is expected to increase accuracy and to decrease cold start and data sparsity problem. The results of CF and CB engines are combined using an appropriate weighted sum function.

In the following subsections, we will discuss the details of the SRDNet.

3.1 Data acquisition

In SRDNet, we collect different types of data to provide an effective recommendation. For this purpose, the movie ratings, the movie features, and the users' social features are employed. We obtained the necessary data from two

Table 2 Overview of deep learning-based movie recommender systems approaches

References	Techniques used	Pros(⊕)/Cons(–)	Dataset
[20]	Marginalized denoising autoencoder Stacked marginalized denoising autoencoder Collaborative filtering	⊕ Solving traditional collaborative filtering problems ⊕ High scalability – Only use collaborative filtering – High computational costs	MovieLens 100K, MovieLens 1M, Book-Crossing, Advertising
[27]	Collaborative filtering Restricted Boltzmann machine Item-based filtering	⊕ Using Restricted Boltzmann Machine (RBM) to learn deeply and predict the ratings or preferences ⊕ Ability to handle sparse dataset – Considering only MAE for evaluation – Not considering changing user's preferences	MovieLens 1M
[28]	MPEG-7 visual descriptor Pre-trained deep-learning neural networks (DNN)	⊕ Visual features extraction from MPEG-7 alongside using deep neural networks – Ignoring social information – Not considering online web application	MovieLens 20M
[29]	Time-aware collaborative filtering Stacked denoising autoencoder	⊕ Link time-aware collaborative filtering and deep learning to solve the sparsity and cold-start problem – Requires extra storage and computation resources – Considering only RMSE for evaluation	Netflix
[44]	Collaborative filtering Autoencoder	⊕ Reducing data sparsity – Ignoring content-based information – Not Considering implicit rating into the autoencoder cost function	MovieLens 100K, MovieLens 1M, R3 Yahoo! Music, MovieTweetings
[45]	End-to-end collaborative filtering with autoencoders	⊕ Linking collaborative filtering and deep learning ⊕ Scalable and robust to deal with large size dataset ⊕ Alleviating the cold start and sparsity – Only use collaborative filtering – Due to privacy issues, collecting personal information is challenging	MovieLens-1M, MovieLens-10M, MovieLens-20M, Douban
[46]	Collaborative filtering Autoencoders	⊕ Link collaborative filtering and deep learning ⊕ Improving representational and computational ability – Only considers the observed ratings in the loss function, which does not guarantee the performance for top-N recommendation – Only use collaborative filtering	MovieLens 1M, MovieLens 10M, Netflix
[47]	Deep neural network using LeakyReLU Integrating the user and item embeddings as features with side information	⊕ Leveraging embeddings, combines side information and a very deep network ⊕ Addressing cold start case and learns of nonlinear latent factors – Focused only on the prediction case and not the ranking case – Inefficiency of the training and recommendation times	MovieLens 100K, MovieLens 1M, FilmTrust
[48]	Nonnegative matrix tri-factorization Stacked denoising autoencoder	⊕ Integrating collective matrix factorization and deep transfer learning – Training nonnegative matrix tri-factorization models on large datasets has computational challenges	MovieLens 100K MovieLens 1M

different sources: MovieTweetings dataset and OMDB web services, as explained below.

The MovieTweetings is a dataset consisting of ratings on movies that were contained in well-structured tweets on Twitter [49], as shown in Table 3. The first purpose of this dataset was to provide the RecSys community with a live, natural, and always updated movie rating dataset from the newest tweets available.

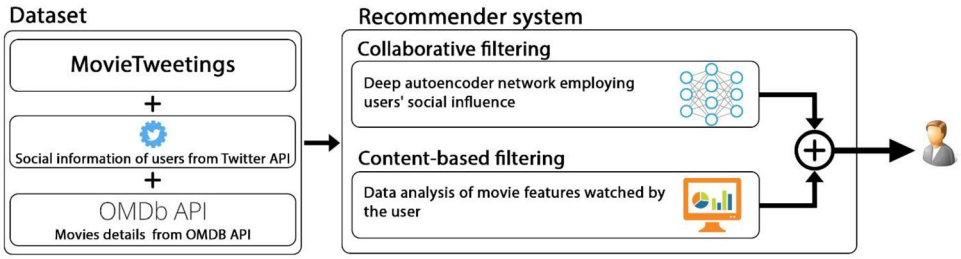

Fig. 1 SRDNet overview

Table 3 Different fields of MovieTweetings dataset

Field	Description	Structure
User ID	User numerical Twitter ID	A nine-digit number
Movie ID	The ID of the rated movie by User in IMDB	A seven-digit number
User rating	User interest toward the Movie	An integer between 0 and 10
Scraping timestamp	A Linux timestamp indicating the moment the Twitter API was queried for the tweet information	Timestamp and date
Name	Movie name and release date	A series of characters and numbers
Genre	Separated movie genres	A series of characters

SRDNet also utilizes a content-based recommender engine, which requires information about the movie details. The main issue of the MovieTweetings dataset is the lack of details about movie features. Therefore, the movie features are extracted employing OMDB web services in JSON format (Table 4).

Table 4 Information extracted for movies from OMDB

Field	Description
Title	Title of movie
Year	Production year
Rating	Movie rating in IMDB
Runtime	The duration of the movie
Genre	Movie genres
Director	Movie director
Writer	Movie writers
Actors	Actors of the movie
Plot	Movie summary
Language	Movie language
Country	Producing country
Awards	Movie awards
Poster	Movie poster URL
Production	Movie producer company
Imdbvotes	Number of IMDB votes
Imdbid	Movie IMDB ID
BoxOffice	Sales in the box office

By integrating above two data sources, the final dataset is obtained, as shown in Table 5.

3.2 Calculating social influence

One of the significant differences between SRDNet and the other approaches available in movie recommendation systems is its social aspects. For this purpose, the social influence of each user is calculated based on his/her social characteristics. In order to obtain the social influence of users, we calculate the average number of Tweets published by user ($\overline{T}$), the average number of retweets ($\overline{R}$), and the average number of Likes ($\overline{F}$) assigned to each tweet, as calculated using Eqs. (5), (6), and (7), respectively.

$$\overline{T} = \frac{M}{N} \tag{5}$$

$$\overline{R} = \frac{1}{M}\sum_{i=1}^{M} Retweet\_count\ T_i \tag{6}$$

$$\overline{F} = \frac{1}{M}\sum_{i=1}^{M} Favorite\_count\ T_i \tag{7}$$

In the above equations, N is the total number of users, M is the total number of tweets, $Retweet\_count\ T_i$ is the number of retweets assigned to tweet T_i, and $Favorite\_count\ T_i$ is the number of likes assigned to tweet T_i (favorite tweets).

If $Tweet\_count_{U_i}$ is considered as the number of tweets published by user U_i, then the average number of expected likes to be assigned to tweets published by user U_i is

Table 5 Features of SRDNet integrated dataset

Metric	Value
Total number of ratings	786,784
Number of unique users	59,712
Number of unique items	34,228
Minimum rating value	0.0
Maximum rating value	10.0
Earliest rating time	2013-02-28 15:38:27
Last rating time	2019-10-11 00:08:33
Maximum number of ratings per user	2821
Maximum number of ratings per item	3066
Number of users with minimum 50 ratings	3578
Number of users with minimum 40 ratings	4465
Number of users with minimum 30 ratings	5828
Number of users with minimum 20 ratings	8084
Number of users with minimum 10 ratings	13,057
Average number of ratings per user	14
Average number of ratings per item	24
Stats calculation time	Fri Oct 11 01:47:46 2019

calculated from Eq. (8), and the average number of retweets expected to be assigned to tweets published by user U_i is obtained from Eq. (9).

$$\overline{FC} = \overline{F} * Tweet\_count_{U_i} \tag{8}$$

$$\overline{RC} = \overline{R} * Tweet\_count_{U_i} \tag{9}$$

Therefore, the social influence of a user is significant when the number of retweets per user's tweets exceeds $\overline{RC}$ and the number of likes of the tweets exceeds $\overline{FC}$. Finally, according to the extracted social data in the dataset and Eqs. (5)–(9), the social influence of each user is calculated using Eq. (10):

$$Social\ influence U_i = \left[\frac{Tweet\_count_{U_i}}{\overline{T}} + \frac{\sum_{i=1}^{Tweet\_count_{U_i}} Favorite\_countT_{U_i}}{\overline{Fc}} + \frac{\sum_{i=1}^{Tweet\_count_{U_i}} Retweet\_countT_{U_i}}{\overline{Rc}} \right] \tag{10}$$

Finally, the sum of averages is rounded up to a nearest higher integer value, and according to that number, the corresponding row of the user is repeated in the matrix user-movie. Therefore, the user ratings and interests that have a greater social influence will have more effect on training the deep autoencoder network. In fact, after creating the user-movie matrix with 59,712 rows (the number of users) and 34,228 columns (the number of movies), the corresponding row of the user is repeated in the matrix according to the number obtained from Eq. (10). Figure 2 shows the effect of the social influence of the user on the user-movie matrix.

3.3 Collaborative filtering

In this phase of SRDNet, user-based collaborative filtering is utilized to predict the ratings that are not present in the dataset by analyzing the ratings of other users who have seen the movies. After predicting the movie ratings, movies with the highest predicted ratings will be recommended to the user.

The user-based collaborative filtering algorithm recommends items with the highest ratings assigned by N users with the most similarities to the target user. The similarities between the target user and other users are usually estimated based on the Pearson correlation coefficient which is one of the most commonly used metrics in the recommender system. The Pearson correlation between users u and v is calculated using Eq. (11):

$$Sim(u,v) = \frac{\sum_{\alpha \in O_{u,v}} \left(r_{u,\alpha} - \overline{r_u}\right)\left(r_{v,\alpha} - \overline{r_v}\right)}{\sqrt{\sum_{\alpha \in O_{u,v}} \left(r_{u,\alpha} - \overline{r_u}\right)^2} \sqrt{\sum_{\alpha \in O_{u,v}} \left(r_{v,\alpha} - \overline{r_v}\right)^2}} \tag{11}$$

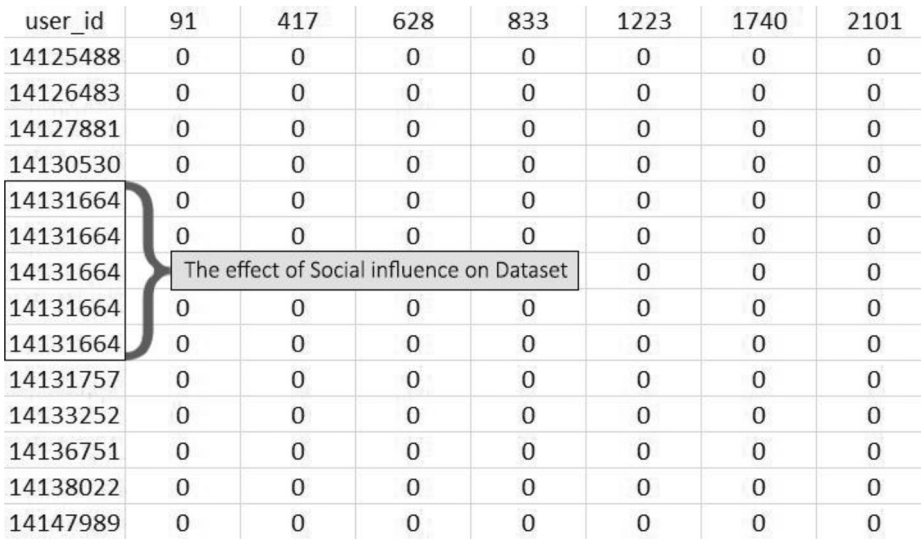

Fig. 2 The effect of the user's social influence on user-item matrix

In the above equation, $r_{u,\alpha}$ is the assigned rating by user u to the movie α and $r_{v,\alpha}$ is the rating assigned by user v to the movie α. $\overline{r_u}$ is the average ratings of user u and $\overline{r_v}$ is the average ratings of user v in the dataset. $O_{u,v}$ set includes the common rated items between user u and v.

After identifying similar users to the target user, in the next step, the unrated items of the target user are predicted to provide accurate recommendations to the target user. Achieving this goal requires precise data analysis and the extraction of complex relations among them. For this purpose, the deep learning approach has been employed in this study.

Among different deep generative models, the deep autoencoder networks propose high accuracy and do not need labeled data. Therefore, they can automatically extract useful features from data. Autoencoders reconstruct data and reduce data dimensions by learning the relations between them.

For developing the deep autoencoder network, the input data format is first prepared to feed to the network. Given the total number of movies, an array with 34,228 elements is formed for each user, which includes the user ratings assigned to each movie. These ratings are integers between 1 and 10. For other movies where the user has not yet scored, this rate is set to 0.

To train the deep autoencoder network, we classify the input data into 80% for training and 20% for testing purposes. As explained in Sect. 3.2, the social influence factors for each user are considered in the training dataset. Therefore, when producing a recommendation, the user who has more social influence will have a greater impact on the final output.

Due to the autoencoder attributes, we first reduce the input data to its feature vector. Then, we try to reconstruct the data from the feature vector. The input layer contains 34,228 nodes, which is equal to the total number of movies. This network has been constructed and evaluated in 5, 7, 9, and 11 layers (the evaluation results are reported in Sect. 4.2). The configuration of each architecture is given in Table 6. In this method, the quality criterion is to compare the nonzero elements from the input record with the same elements in the output record.

It should be noted that the number of epochs and the batch size has been considered 500 and 100 during the training, respectively. This means that the entire training set will feed to the autoencoder 500 times, every time using 100 users. The learning rate is set to 0.001. The sigmoid and MSE have been utilized as activation and loss functions, respectively. Figure 3 shows the 7-layer autoencoder network architecture. As mentioned before, the input layer contains 34,228 nodes equal to the total number of movies and an array of user ¡ ratings to the movies.

As shown in the above figure, in autoencoder-based collaborative filtering, the number of neurons in the input layer is the same as the number of movies in the dataset. After each forward propagation, the error of each output neuron has been calculated and backpropagated to the input layer to minimize the total error. The final results should predict the user's preferences for his/her unseen movies. The following is a 7-layer autoencoder-based collaborative filtering pseudo-code.

Algorithm 1: Autoencoder-based collaborative filtering pseudo-code

```
Input: User_i  Ratings
Output: Top 10 Movie IDs that have the highest values in the reconstructed user's rating vector
 1: S = Calculate_Social_influence()
 2: Repeat_rows(S)
 3: For user_i ∈ Users do
 4:     Rates=row(i)
 5:     Recommendation_Set = []
 6: end for
 7: batchsize = 100
 8: epoch_num = 500
 9: Setting_the_Model_Parameters()
10: Calculate similarity between user_i and other users by Pearson correlation measure
11: Input_Processing(row(i))
12: row'(i)=Reconstruction(row(i))
13: Calculate_MAE()
```

3.4 Content-based filtering

In this phase of SRDNet, the movie recommendation is generated according to two factors. These two factors include the ratings by the target user and the movie information. This part of the recommendation engine is more effective for users with fewer movie ratings than the average number of movies rated by each user, and its main task is to overcome the cold start problem. However, this engine is also applicable to users with more ratings than the average number of movies rated by each user. The accuracy of SRDNet increases employing this recommendation engine along with the collaborative filtering.

Content-based filtering recommendation engine works as follows: first, the history of movies watched by the target user is examined, and they are grouped together (M set in Algorithm 2). Then, the list of movies from this set that has been rated by user 7 or higher is marked as his/her favorite movies (F set in Algorithm 2).

In the next step, all features of these movies are extracted from the dataset, and four sets of actors, writers, directors, and genres are created (A, D, W, and G sets in Algorithm 2). Then, based on each set, some movies are

Table 6 Configuration of deep autoencoder network for different number of layers

Number of layers	Groups of neurons in each layer
5	34,228–10,000–500–10,000–34,228
7	34,228–10,000–5000–500–5000–10,000–34,228
9	34,228–10,000–5000–1000–500–1000–5000–10,000–34,228
11	34,228–12,000–10,000–5000–1000–500–1000–5000–10,000–12,000–34,228

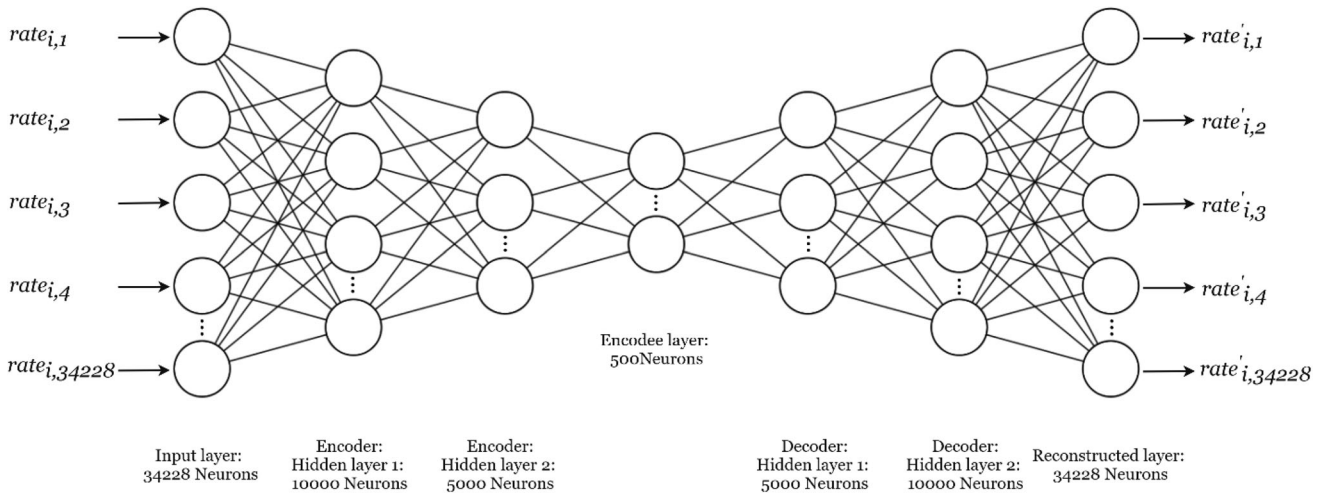

Fig. 3 7-layer deep autoencoder network architecture

suggested to the target user so that each movie has one of the characteristics that the target user has watched a movie with the same characteristics in the past.

For the sake of accuracy, movies that belong to all sets are placed at the top of the recommendation list. If the result of the intersection of all sets is null, then movies with higher ratings in the union of all sets will be recommended to the target user. Below pseudo-code shows the content-based filtering in SRDNet.

Algorithm 2: Pseudo-code of content-based filtering in SRDNet

```
Input: M: Movie IDs that User_i has already seen
Output: Top 10 Movie IDs
 1: List F, Rec;
 2: F = movies from M that have been rated 7 or above
 3: Rec = Ø
 4: For each Movie ID of F do
 5:     List A, D, W, G, L1, L2, L3, L4;
 6:     A = Actors who play a role in movies in F
 7:     L1 = Movies that Actors in A play a role in that
 8:     Rec.add(L1);
 9:     D =   Directors of movies in F
10:     L2 =   Movies that have been directed by Directors in D
11:     Rec.add(L2);
12:     W = Writers of movies in F
13:     L3 = movies that have been written by writers in W
14:     Rec.add(L3);
15:     G = Genres of movies in F
16:     L4 = movies that are in G genre
17:     Rec.add(L4);
18: End for
19: Return Top(10,Rec)
```

3.5 Combination of the recommendations

Considering that SRDNet utilizes two recommendation engines, it benefits from the characteristics of each of them. The weighting factor of each recommendation engine varies according to the number of the ratings of the user. In general, the combination of recommendation engines is formulated by Eq. (12):

$$Final_{rs} = \alpha CF_{rs} + \beta CB_{rs} \quad \text{where: } \alpha + \beta = 1 \tag{12}$$

This equation represents that the final list of recommendations is a combination of two content-based filtering (CB_{rs}) and the collaborative filtering (CF_{rs}) recommendations. The values of α and β are the weighting coefficients of each set in the final recommendation list and are provided in Table 7. In fact, the values of α and β are chosen based on the number of user-rated movies; for example, if a user has watched and rated two movies, α and β will be considered 0.2 and 0.8, respectively. This value is set based on the average number of ratings per user in the dataset, which is equal to 14. In fact, for users who have watched and rated more movies, collaborative filtering needs to have more weight to utilize other users' interests to generate appropriate recommendations.

However, this is also obvious by examining the statistics of the rated movies by users in the dataset. In fact, the proposed dataset suffers from the cold start problem, like other real-world scenarios where many users have not rated a significant set of movies. In our dataset, the average number of ratings per user is 14. Therefore, the focus of the collaborative filtering recommender engine should be on users who have rated fewer movies, so the weight of α in this interval is divided in such a way that, in addition to minimizing the effect of the cold start, it increases the accuracy of recommendations for users who have rated more movies.

Table 7 The weight coefficients distribution between two recommendation engines in SRDNet

Number of movies rated by the target user	α	β
0–1	0.1	0.9
2–3	0.2	0.8
4–5	0.3	0.7
6–7	0.4	0.6
8–9	0.5	0.5
10–11	0.6	0.4
12–13	0.7	0.3
14	0.8	0.2
15 and more	0.9	0.1

4 Simulations and evaluation

In this section, we conduct different experiments to demonstrate the effectiveness of SRDNet. For this purpose, we first introduce the evaluation metrics. Then, the experiments and the related results of SRDNet are reported. Finally, the results are analyzed and compared with the other state-of-the-art methods. Python programming language version 3.5, and the TensorFlow library has been employed to implement SRDNet. TensorFlow which is an open-source platform for machine learning has been utilized to develop the deep autoencoder networks. In terms of hardware, SRDNet has been deployed on a system with an Intel Core i7-6700K, 16 GB memory, and a GeForce GTX 1060 6 GB graphic card with 1280 CUDA cores.

4.1 Evaluation metrics

According to [1–3], the prediction error is directly related to the recommendation quality. Therefore, the less prediction error rate will increase the effectiveness of recommendations. In order to evaluate the results of a recommender system, some standard error prediction metrics are used, among which the mean absolute error (MAE) and root mean squared error (RMSE) are used for SRDNet evaluation. In addition, Spearman rank correlation coefficient is used to calculate the accuracy of estimated ratings.

We define U as a set of recommender system's users, I as a set of recommender system's items, $r_{u,i}$ the rating of user u to item i, Θ as no rating. $r_{u,i} \neq \Theta$ means user u has not assigned any rating to item i and $P_{u,i}$ is the predicted rating of user u to item i. If $O_u = \{i \epsilon I | P_{u,i} \neq \Theta \wedge r_{u,i} \neq \Theta\}$ is a set of rated items by user u with predicted ratings, also, the absolute difference between the predicted and actual values, $|P_{u,i} - r_{u,i}|$, shows the prediction error. Therefore, MAE and RMSE are introduced based on Eqs. (13) and (14):

$$\mathrm{MAE} = \frac{1}{\#U}\left(\frac{1}{\#O_u}\sum_{i\in O_u}|P_{u,i} - r_{u,i}|\right) \tag{13}$$

$$\mathrm{RMSE} = \frac{1}{\#U}\sum_{u\in U}\sqrt{\frac{1}{\#O_u}\sum_{i\in O_u}(P_{u,i} - r_{u,i})^2} \tag{14}$$

In (13) and (14), $\#O_u$ represents the number of members in O_u set and $\#U$ shows the number of users in the recommender system.

Spearman rank correlation is used to calculate the accuracy of the estimated ratings, which is used when the data are sequentially discontinuous (1, 2, 3, …), or the values are converted to a rank. If the data are measured by distance or relative scale, they can be converted to ranks and then calculated by the Spearman rank correlation coefficient. Spearman rank correlation coefficient which is represented by r is calculated according to (15):

$$r = 1 - \frac{6}{\#O_u(\#O_u^2 - 1)}\sum_{i=1}^{\#O_u}(d_i)^2 \tag{15}$$

In Eq. (15), d_i is the difference between the predicted rate and the actual rate of items in O_u set and $\#O_u$ represents the number of members in O_u dataset.

4.2 Evaluation of SRDNet

As mentioned in Sect. 3.3, a deep autoencoder network is used to predict unavailable ratings in SRDNet. The prediction accuracy has been assessed with the number of layers in Figs. 4 and 5 based on MAE and RMSE metrics.

As depicted in Figs. 4 and 5, both MAE and RMSE metrics are at the lowest value in the 7-layer autoencoder network, and when the number of layers increases, there is no more decrease in error.

Figure 6 shows the rank metric in the deep autoencoder network in two phases of training and test for 5, 7, 9, and

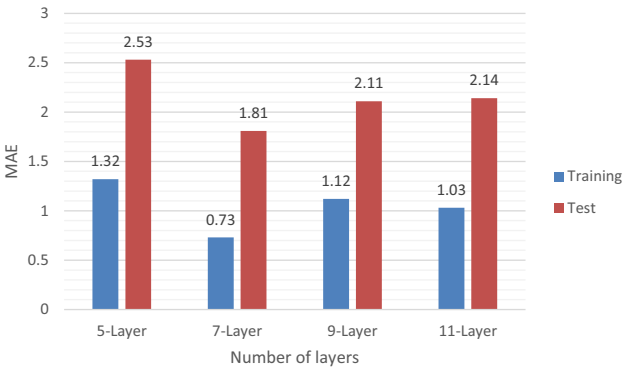

Fig. 4 MAE metric versus number of layers in SRDNet deep autoencoder network

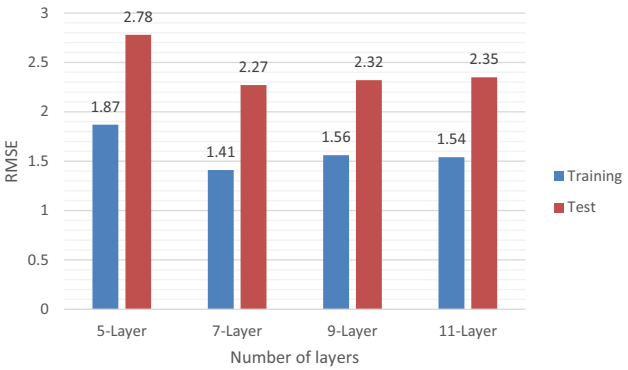

Fig. 5 RMSE metric versus number of layers in SRDNet deep autoencoder network

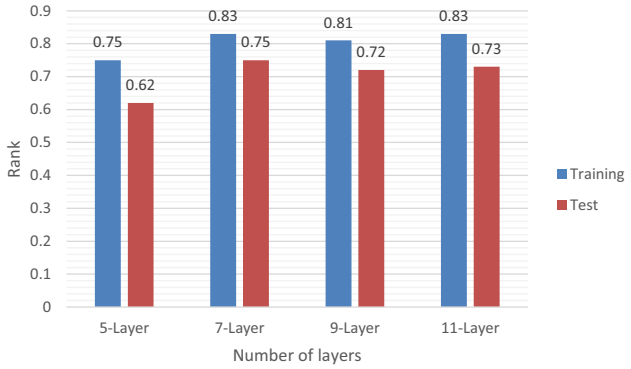

Fig. 6 Rank metric versus number of layers of SRDNet deep autoencoder network

11 layers. In evaluating the quality of the recommendations by ranking metric, the 7-layer architecture in the autoencoder network shows a better performance in both the training and test phase.

Table 8 summarizes the performance results of the deep autoencoder network in SRDNet for a different number of layers.

4.3 SRDNet comparison

In this section, we assess SRDNet with the other state-of-the-art methods in movie recommender systems. Since SRDNet is a hybrid approach based on CF and CB utilizing a deep neural network, several approaches have been selected for the comparison purpose that employs different machine learning (ML), neural network (NN), or deep neural network (DNN). To this end, five methods are based on CF that have been classified into DNN-based approaches (MRS-RBM [27], A-COFILS [44], and AutoRec [46]) and ML-based approaches (PP-CF [50] and VB-CF [51]). Three other methods are based on CB that have been utilized DNN (MV-DNN [52]), ML (MML-CB [53]), and NN (FCMR [54]).

The results of the evaluations are based on the MovieTweetings dataset. MovieTweetings is the only movie recommendation system dataset that provides information about the user's social networks (Twitter). Using this information, the social influence of users can be calculated. Indeed, this dataset includes Twitter IDs of the users, so we can extract the user's social information from Twitter using Twitter API.

Behera et al. presented a hybrid movie recommender system based on CF and deep learning employed restricted Boltzmann machine (RBM) to predict not available ratings in the dataset [27]. RBM estimates the interest of the target user to the unseen movies based on his/her previous ratings to other movies.

The first approach for the comparison purpose is autoencoder COFILS (A-COFILS) presented by Barbieri et al. [44]. This approach focuses on reducing the effect of the noisy data on the results as well as increasing the ability of the proposed method to extract complex nonlinear features. Barbieri et al. employed stacked denoising autoencoders (SDA) instead of the singular value decomposition (SVD) in the first phase of COFILS, which leads to learning more useful and complex representations in neural networks.

Sedhain et al. utilized an autoencoder deep network to find unknown values of users' ratings [46]. This approach is autoencoder-based collaborative filtering approach that is capable of learning nonlinear data representations.

Polatidis et al. proposed a multi-level privacy-preserving method for collaborative filtering (PP-CF) systems by perturbing each rating before it was submitted to the recommender server [50]. For this purpose, they utilized a learning model for the perturbation algorithm which can easily be installed on the client-side.

Langseth et al. focused on the probabilistic collaborative filtering that explicitly represents all users and items simultaneously in the model [51]. To solve the poor scalability of the probabilistic CF, they proposed an approximate learning and inference algorithms based on a variational Bayes approach. The algorithm employs a mean-field approximation of the variational distribution to ensure that the complexity of the learning algorithm grows linearly.

Elkahky et al. used a deep neural network on the manually extracted user and item feature representations for content-based multi-domain recommendation [52]. For this purpose, they introduced a multi-view deep neural network (MV-DNN) to build recommendation systems by combining datasets from multiple domains. MV-DNN employs an extension to the DSSM where more than two views of

Table 8 SRDNet deep autoencoder network performance for a different number of layers

Layers	Training				Test			
	Time (s)	MAE	RMSE	Rank	Time (s)	MAE	RMSE	Rank
5	136	1.32	1.87	0.75	17	2.53	2.78	0.62
7	193	0.73	1.41	0.83	20	1.81	2.27	0.75
9	270	1.12	1.56	0.81	22	2.11	2.32	0.72
11	420	1.03	1.54	0.83	31	2.14	2.35	0.73

data are utilized. DSSM is a deep structured semantic model to enhance query document matching in the web search context. MV-DNN maps high-dimensional sparse movie features into low-dimensional dense features in a joint semantic space.

Soares et al. proposed multimedia (films and television programs) recommender system that utilizes a set of metadata elements that include the title, a genre, the date of production, and the list of directors and actors [53]. The authors demonstrated that the use of different metadata elements can contribute to increase the quality of the recommendations. They employed two different CB approaches, namely "nearest neighbor" and "genre learning process".

Chen et al. employed the movie content information (genres, cast, crew, etc.) to obtain a vector form features of each element and then take advantage of the linear relationship of the learned features to calculate the similarity between each movie [54]. For this purpose, the authors utilized a Word2vec model using the movie content information as the training data. Word2vec is used to produce word embeddings based on a shallow neural network model. The system recommends movies to the users according to the similarities between movies.

The comparison results of SRDNet with the above methods are mentioned in Table 9 and also depicted in Figs. 7 and 8. The evaluation results indicate that SRDNet shows a better performance than other methods in MAE and RMSE metrics.

The results of the approach that utilizes the combination of CB and DNN (MV-DNN) are better than CB plus ML or NN (MML-CB and FCMR). The approaches based on CF and DNN (A-COFILS, AutoRec, and MRS-RBM) also outperform methods based on CF and ML (PP-CF and VB-CF). These results are quite reasonable, given the fact that DNN-based approaches usually provide highly accurate results.

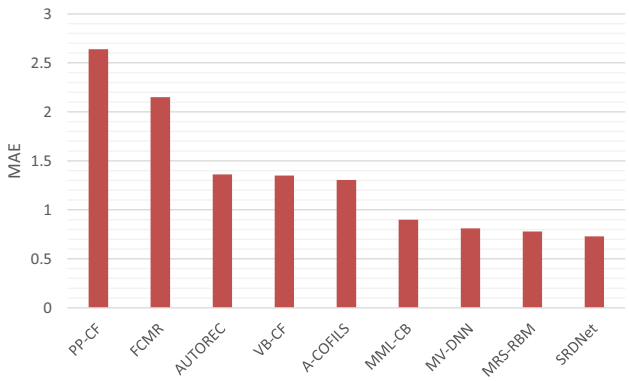

Fig. 7 MAE comparison results

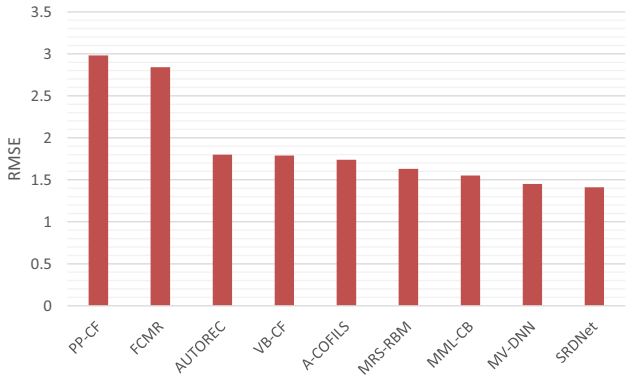

Fig. 8 RMSE comparison results

However, the accuracy results of these methods are still lower than the SRDNet. Indeed, these methods only use one recommender engine (CF or CB), and the ratings/interests of the most influential users do not contribute to the training of the DNN. As explained before, SRDNet enjoys CF and CB at the same time as well as a deep autoencoder network to reduce the prediction error, and consequently, the accuracy of recommendations is increased. Moreover, SRDNet employs the concept of social influence of users,

Table 9 MAE/RMSE results of different approaches based on MovieTweetings dataset

Approach	Method	Description	MAE	RMSE
CF + CB + DNN	SRDNet	CF + CB + deep autoencoder network	0.73	1.41
CF + DNN	MRS-RBM [27]	CF + restricted Boltzmann machine (RBM)	0.78	1.63
	A-COFILS [44]	CF + stacked denoising autoencoders (SDA)	1.30	1.74
	AutoRec [46]	CF + deep autoencoder network	1.36	1.80
CF + ML	PP-CF [50]	CF + perturbation learning model	2.64	2.98
	VB-CF [51]	CF + variational Bayes	1.35	1.79
CB + DNN	MV-DNN [52]	CB + multi-view deep neural network	0.81	1.45
CB + ML	MML-CB [53]	CB + nearest neighbor and a genre learning process	0.90	1.55
CB + NN	FCMR [54]	CB + Word2vec (shallow neural network)	2.15	2.84

which is calculated based on their social activities on Twitter. In order to train the deep autoencoder network, the ratings and interests of the user with more social influence will be more effective.

5 Conclusion and future works

In this paper, we proposed a novel recommendation approach called SRDNet that utilizes a deep autoencoder network and the concept of social influence. SRDNet is a hybrid social movie recommender system that benefits from both collaborative and content-based filtering techniques to generate high-accuracy recommendations. Moreover, the social influence of each user is calculated based on the extracted social information of users. Therefore, users with higher social influence impose a greater influence on the recommended movies to the target users. Then, the unrated movies of target users have been predicted using a deep autoencoder network.

In SRDNet, after calculating user's social influence, the user-based collaborative filtering algorithm identifies similar users to the target user and predicts the unrated items based on a 7-layer deep autoencoder. The content-based filtering is item based, which recommends movies to users by analyzing movie features. Then, the results of collaborative filtering and content-based filtering are combined using an appropriate weighted sum function. By combining these two filtering algorithms, accuracy is increased, and cold start and data sparsity problems are decreased.

For evaluation purposes, the required dataset has been collected from MovieTweetings and OMDB. The results of evaluations and comparisons of SRDNet with other state-of-the-art methods show that the proposed approach reduces both MAE and RMSE.

As future work, it is suggested to consider below items:

- Increasing the use of social data will provide a better view of users' preferences and interests; so utilizing the sentiment analysis and natural language processing of tweets can increase the accuracy and effectiveness of the generated recommendations.
- Investigating user relationships in social networks such as following and followers can improve the results.
- Using deep learning in analyzing the frame-by-frame of movies or exploring movie posters to identify movie genres and subjects provides us valuable feature sets that can be employed in the movie recommendation process to increase the diversity and novelty (serendipity concept).

References

1. Bobadilla J, Ortega F, Hernando A, Gutiérrez A (2013) Recommender systems survey. Knowl Based Syst 46:109–132
2. Kunaver M, Požrl T (2017) Diversity in recommender systems—a survey. Knowl Based Syst 123:154–162
3. Yang X, Guo Y, Liu Y, Steck H (2014) A survey of collaborative filtering based social recommender systems. Comput Commun 41:1–10
4. Mochón M-C (2016) Social network analysis and big data tools applied to the systemic risk supervision. Int J Interactive Multimed Artif Intell 3(6):34–37
5. Lies J (2019) Marketing intelligence and Big Data: digital Marketing techniques on their way to becoming social engineering techniques in marketing. Int J Interactive Multimed Artif Intell 5(5):134–144
6. Crespo RG, Martínez OS, Lovelle JMC, García-Bustelo BCP, Gayo JEL, De Pablos PO (2011) Recommendation system based on user interaction data applied to intelligent electronic books. Comput Hum Behav 27(4):1445–1449
7. Li J, Xu W, Wan W, Sun J (2018) Movie recommendation based on bridging movie feature and user interest. J Comput Sci 26:128–134
8. Guzmán de Núñez X, Núñez Valdéz ER, Pascual Espada J, González Crespo R, García Díaz V (2018) A proposal for sentiment analysis on twitter for tourism-based applications. In: Fujita H, Herrera-Viedma E (eds) New trends in intelligent software methodologies, tools and techniques. IOS Press, Amsterdam, pp 713–722
9. Guy I (2015) Social recommender systems. In: Kantor PB, Ricci F, Shapira B, Rokach L (eds) Recommender systems handbook. Springer, Berlin, pp 511–543
10. Zhang S, Yao L, Sun A, Tay Y (2019) Deep learning based recommender system: a survey and new perspectives. ACM Comput Surv (CSUR) 52(1):1–38
11. Ouyang Y, Liu W, Rong W, Xiong Z (2014) Autoencoder-based collaborative filtering. In: International conference on neural information processing. Springer, Berlin, pp 284–291
12. Chen Y, de Rijke M (2018) A collective variational autoencoder for top-n recommendation with side information. In: Proceedings of the 3rd workshop on deep learning for recommender systems. ACM, pp 3–9
13. Li X, She J (2017) Collaborative variational autoencoder for recommender systems. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 305–314
14. Liang D, Krishnan RG, Hoffman MD, Jebara T (2018) Variational autoencoders for collaborative filtering. In: Proceedings of the 2018 world wide web conference, 2018. International World Wide Web Conferences Steering Committee, pp 689–698
15. Jhamb Y, Ebesu T, Fang Y (2018) Attentive contextual denoising autoencoder for recommendation. In: Proceedings of the 2018 ACM SIGIR international conference on theory of information retrieval. ACM, pp 27–34
16. Wang M, Wu Z, Sun X, Feng G, Zhang B (2019) Trust-aware collaborative filtering with a denoising autoencoder. Neural Process Lett 49(2):835–849
17. Wang K, Xu L, Huang L, Wang C-D, Lai J-H (2019) SDDRS: stacked discriminative denoising auto-encoder based recommender system. Cogn Syst Res 55:164–174
18. Wu Y, DuBois C, Zheng AX, Ester M (2016) Collaborative denoising auto-encoders for top-n recommender systems. In: Proceedings of the ninth ACM international conference on web search and data mining. ACM, pp 153–162

19. Li S, Fu Y (2017) Robust representations for collaborative filtering. Robust representation for data analytics. Springer, Berlin, pp 123–146
20. Li S, Kawale J, Fu Y (2015) Deep collaborative filtering via marginalized denoising auto-encoder. In: Proceedings of the 24th ACM international on conference on information and knowledge management. ACM, pp 811–820
21. Rifai S, Vincent P, Muller X, Glorot X, Bengio Y (2011) Contractive auto-encoders: explicit invariance during feature extraction. In: Proceedings of the 28th international conference on international conference on machine learning. Omnipress, pp 833–840
22. Zhang S, Yao L, Xu X (2017) Autosvd++: an efficient hybrid collaborative filtering model via contractive auto-encoders. In: Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval. ACM, pp 957–960
23. Capdevila J, Arias M, Arratia A (2016) GeoSRS: a hybrid social recommender system for geolocated data. Inf Syst 57:111–128
24. López-Quintero JF, Lovelle JC, Crespo RG, García-Díaz V (2018) A personal knowledge management metamodel based on semantic analysis and social information. Soft Comput 22(6):1845–1854
25. Das N, Borra S, Dey N, Borah S (2018) Social networking in web based movie recommendation system. In: Dey N, Babo R, Ashour AS, Bhatnagar V, Bouhlel MS (eds) Social networks science: design, implementation, security, and challenges. Springer, Berlin, pp 25–45
26. Li F, Xu G, Cao L (2016) Two-level matrix factorization for recommender systems. Neural Comput Appl 27(8):2267–2278
27. Behera DK, Das M, Swetanisha S (2019) Predicting users' preferences for movie recommender system using restricted Boltzmann machine. In: Della Riccia G, Kruse R, Lenz H-J (eds) Computational intelligence in data mining. Springer, Berlin, pp 759–769
28. Deldjoo Y, Elahi M, Quadrana M, Cremonesi P (2018) Using visual features based on MPEG-7 and deep learning for movie recommendation. Int J Multimed Inf Retrieval 7(4):207–219
29. Wei J, He J, Chen K, Zhou Y, Tang Z (2017) Collaborative filtering and deep learning based recommendation system for cold start items. Expert Syst Appl 69:29–39
30. Pattanayak S (2017) Unsupervised learning with restricted Boltzmann machines and auto-encoders. Pro deep learning with tensorflow. Springer, Berlin, pp 279–343
31. Sun Y, Mao H, Sang Y, Yi Z (2017) Explicit guiding auto-encoders for learning meaningful representation. Neural Comput Appl 28(3):429–436
32. Sun Y, Mao H, Guo Q, Yi Z (2016) Learning a good representation with unsymmetrical auto-encoder. Neural Comput Appl 27(5):1361–1367
33. Bengio Y, Courville A, Vincent P (2013) Representation learning: a review and new perspectives. IEEE Trans Pattern Anal Mach Intell 35(8):1798–1828
34. Covington P, Adams J, Sargin E (2016) Deep neural networks for youtube recommendations. In: Proceedings of the 10th ACM conference on recommender systems. ACM, pp 191–198
35. Cheng H-T, Koc L, Harmsen J, Shaked T, Chandra T, Aradhye H, Anderson G, Corrado G, Chai W, Ispir M (2016) Wide & deep learning for recommender systems. In: Proceedings of the 1st workshop on deep learning for recommender systems. ACM, pp 7–10
36. Okura S, Tagami Y, Ono S, Tajima A (2017) Embedding-based news recommendation for millions of users. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, ACM, pp 1933–1942
37. Li H, Cui J, Shen B, Ma J (2016) An intelligent movie recommendation system through group-level sentiment analysis in microblogs. Neurocomputing 210:164–173
38. Sun Z, Han L, Huang W, Wang X, Zeng X, Wang M, Yan H (2015) Recommender systems based on social networks. J Syst Softw 99:109–119
39. Seo Y-D, Kim Y-G, Lee E, Baik D-K (2017) Personalized recommender system based on friendship strength in social network services. Expert Syst Appl 69:135–148
40. Zhao Z, Yang Q, Lu H, Weninger T, Cai D, He X, Zhuang Y (2017) Social-aware movie recommendation via multimodal network learning. IEEE Trans Multimed 20(2):430–440
41. Pérez-Marcos J, Martín-Gómez L, Jiménez-Bravo DM, López VF, Moreno-García MN (2020) Hybrid system for video game recommendation based on implicit ratings and social networks. J Ambient Intell Humaniz Comput. https://doi.org/10.1007/s12652-020-01681-0
42. Katarya R, Verma OP (2018) Recommender system with grey wolf optimizer and FCM. Neural Comput Appl 30(5):1679–1687
43. Ling Z, Xiao Y, Wang H, Xu L, Hsu C-H (2019) Extracting implicit friends from heterogeneous information network for social recommendation. In: Pacific Rim international conference on artificial intelligence. Springer, pp 607–620
44. Barbieri J, Alvim LG, Braida F, Zimbrão G (2017) Autoencoders and recommender systems: COFILS approach. Expert Syst Appl 89:81–90
45. Strub F, Gaudel R, Mary J (2016) Hybrid recommender system based on autoencoders. In: Proceedings of the 1st workshop on deep learning for recommender systems. ACM, pp 11–16
46. Sedhain S, Menon AK, Sanner S, Xie L (2015) Autorec: autoencoders meet collaborative filtering. In: Proceedings of the 24th international conference on world wide web. ACM, pp 111–112
47. Kiran R, Kumar P, Bhasker B (2020) DNNRec: a novel deep learning based hybrid recommender system. Expert Syst Appl 144:113054
48. Gai S, Zhao F, Kang Y, Chen Z, Wang D, Tang A (2019) Deep transfer collaborative filtering for recommender systems. In: Pacific Rim international conference on artificial intelligence. Springer, pp 515–528
49. Dooms S, De Pessemier T, Martens L (2013) Movietweetings: a movie rating dataset collected from twitter. In: Workshop on crowdsourcing and human computation for recommender systems, CrowdRec at RecSys, p 43
50. Polatidis N, Georgiadis CK, Pimenidis E, Mouratidis H (2017) Privacy-preserving collaborative recommendations based on random perturbations. Expert Syst Appl 71:18–25
51. Langseth H, Nielsen TD (2015) Scalable learning of probabilistic latent models for collaborative filtering. Decis Support Syst 74:1–11
52. Elkahky AM, Song Y, He X (2015) A multi-view deep learning approach for cross domain user modeling in recommendation systems. In: Proceedings of the 24th international conference on world wide web, pp 278–288
53. Soares M, Viana P (2015) Tuning metadata for better movie content-based recommendation systems. Multimed Tools Appl 74(17):7015–7036
54. Chen H-W, Wu Y-L, Hor M-K, Tang C-Y (2017) Fully content-based movie recommender system with feature extraction using neural network. In: 2017 international conference on machine learning and cybernetics (ICMLC). IEEE, pp 504–509