Intelligent Automation & Soft Computing
DOI: 10.32604/iasc.2023.030361
Article

Tech Science Press

Content-Based Movie Recommendation System Using MBO with DBN

S. Sridhar[1,*], D. Dhanasekaran[2] and G. Charlyn Pushpa Latha[3]

[1]Department of CSE, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, India
[2]Saveetha Institute of Medical and Technical Sciences, Chennai, India
[3]Department of IT, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai, India
*Corresponding Author: S. Sridhar. Email: 007sridol@gmail.com
Received: 24 March 2022; Accepted: 26 April 2022

Abstract: The content-based filtering technique has been used effectively in a variety of Recommender Systems (RS). The user explicitly or implicitly provides data in the Content-Based Recommender System. The system collects this data and creates a profile for all the users, and the recommendation is generated by the user profile. The recommendation generated via content-based filtering is provided by observing just a single user's profile. The primary objective of this RS is to recommend a list of movies based on the user's preferences. A content-based movie recommendation model is proposed in this research, which recommends movies based on the user's profile from the Facebook platform. The recommendation system is built with a hybrid model that combines the Monarch Butterfly Optimization (MBO) with the Deep Belief Network (DBN). For feature selection, the MBO is utilized, while DBN is used for classification. The datasets used in the experiment are collected from Facebook and MovieLens. The dataset features are evaluated for performance evaluation to validate if data with various attributes can solve the matching recommendations. Each file is compared with features that prove the features will support movie recommendations. The proposed model's mean absolute error (MAE) and root-mean-square error (RMSE) values are 0.716 and 0.915, and its precision and recall are 97.35 and 96.60 percent, respectively. Extensive tests have demonstrated the advantages of the proposed method in terms of MAE, RMSE, Precision, and Recall compared to state-of-the-art algorithms such as Fuzzy C-means with Bat algorithm (FCM-BAT), Collaborative filtering with k-NN and the normalized discounted cumulative gain method (CF-kNN+NDCG), User profile correlation-based similarity (UPCSim), and Deep Autoencoder.

Keywords: Movie recommendation; monarch butterfly optimization; deep belief network; facebook; movielens; deep learning

1 Introduction

Recommendation System is an important topic that is extremely popular and beneficial for individuals to make automatic decisions accurately. It is a technology that assists users in extracting useful information

from a large amount of accessible data. Relating to Movie Recommendation System, the recommendation is generated based on user similarities (Collaborative Filtering) or by taking into account a specific user's behaviour (Content-Based Filtering) with which they wish to undertake. A recommendation system is a data tool that assists users in determining the things they desire from a vast number of accessible items [1].

The main aim of this recommendation system is to recommend a list of movies depending on the user's preferences. It assists the user in selecting the finest option from the selection of movies offered. Many platforms, like Netflix, YouTube, and Amazon, utilize recommendation algorithms to serve their customers and increase profits. Many platforms recommend movies, Amazon recommends products, Spotify recommends music, LinkedIn recommends jobs, and every social networking site recommends people. All of this is based on the recommendation system. Users may quickly find out what they want based on their preferences using these recommendation engines. As a result, developing an efficient recommender system is a problem since user preferences change over time [2,3].

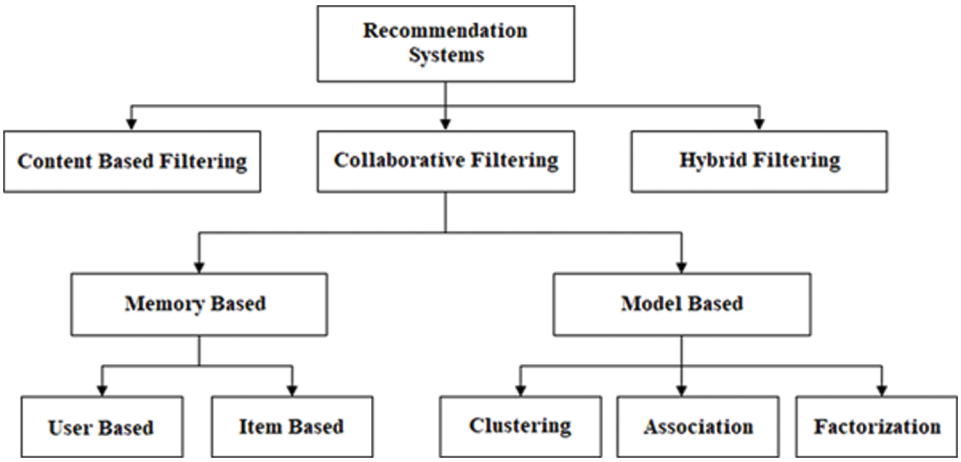

Figure 1: Types of recommendation system

There are three recommendation system techniques (refer to Fig. 1): collaborative filtering, content-based filtering, and hybrid filtering [4]. Cognitive filtering is another name for content-based filtering. In a Content-Based Recommender System, the user either actively or unintentionally gives data (rating or assessing a link). The RS collects this data and makes a profile for all the users. The recommendation was created by utilizing the user profile. The recommendation by content-based filtering is generated by just observing a single user's profile.

Collaborative filtering is a method for filtering data and providing relevant information to the user. One of the most well-known strategies for recommending items is collaborative filtering. Collaborative filtering, as opposed to content-based filtering, discovers people whose preferences are equal to those of a particular user. Therefore, it recommends a product or any item based on the likelihood that the given user would further enjoy the thing, which other users enjoy since their interests are identical. Hybrid filtering may be utilized in a variety of ways. It addresses cold start, data sparsity, and scalability issues. The system can employ content-based filtering initially and thus transfer the results to collaborative filtering (and vice versa), or it could integrate both filters into a single system for better outcomes [5,6].

In this research, the main focus of the work was to propose a movie recommendation model, which recommends movies based on the user's profile from the Facebook platform. The proposed model is developed using a hybrid model that includes the MBO and DBN. The MBO is used for feature

selection, and DBN is used for classification. The proposed model is based on a content-based filtering technique.

The scope of this research is to recommend movies by developing a content-based recommender system, which the system helps E-commerce sites to increase their sales. A movie recommendation system based on a content-based filtering approach uses the information provided by users, analyzes them and then recommends the movie that is best suited to the user from their friend's group. This proposed RS can be used on social media platforms like Facebook Watch and YouTube.

The remaining parts of this work are as follows: Section 2 discusses the literature review based on related works, Section 3 presents the proposed methodology, Section 4 presents the experimental analysis, and Section 5 presents the conclusion and future scope of the work.

2 Related Works

Ashish et al. developed a recommendation system based on a modification of the whale optimization algorithm, with tournament selection enabling the whale optimization method to find the best clusters. On the publicly accessible movie-lens dataset, this technique was evaluated as a recommendation system. According to the experimental results, this system was a permissive approach for a recommendation over large-scale datasets. This tournament selection approach was ineffective in identifying the top solutions [7].

Subramaniyaswamy et al. used filtering and clustering techniques to recommend movies of interest to users. Initially, user preference was collected by allowing people to rank their favourite movies. Upon the ratings, the recommender system better understood the user and recommended movies that were more likely to be rated higher. Based on the findings, several modifications may be made to increase performance and produce more accurate recommendations [8]. Ramzan et al. developed a new collaborative filtering recommendation technique in which feature matrix by polarity identification was achieved using opinion-based sentiment analysis. This technique integrated syntax analysis, lexical analysis, and semantic analysis to understand sentiment toward features and user type profiling. This system used the big data Hadoop platform to manage heterogeneous data and offered recommendations depending on user type using fuzzy rules. This model achieved an average performance of 74% accuracy, which might be improved [9]. Reddy et al. proposed a movie RS according to the genres, which the user would be interesting to see. The methodology utilized was a content-based filtering approach with genre correlations. All of the frameworks utilized in this analysis have advantages and disadvantages, but they were still insufficient to solve all security, energy, and user experience problems [10].

Subramaniyaswamy et al. created a novel ontology for helping people to comprehend the movie domain better. The user data, which included movie ratings, was utilized for recommending movies to people. The adaptive k-NN (AKNN) technique and post-classifications procedure were utilized for user categorization, and movies were suggested to the active target users. Furthermore, regardless of the datasets utilized, the learning effectiveness of the recommendation technique must be improved [11].

Phonexay et al. provided a solution that solved the cold-start problem while providing better satisfaction than the other approaches utilized in the recommendation system. With the support of the k-clique, this technique classified users into different groups and provided movie suggestions for the user based on their personal information. The k-clique approach might be tweaked to reduce experiment duration and applied to a larger dataset [12].

Zhenning et al. developed a weighted classification and users collaborative filtering approach-based hybrid movies recommendation system and optimization technique. The fundamental recommendation model was the sparse linear model, while the local recommendation model was trained using user clustering. Combining the top-N customized movie recommendation model with the weighted

classification model, the top-N personalized movie recommendation was accomplished [13]. Debajit et al. proposed a work that addressed both Content-Based Filtering and Collaborative to create a product and movie RS for social networking sites that demonstrated the effectiveness of collaborative filtering and depicted the challenges faced by content-based filtering. This technique solved the cold start problem, but it was expensive in relation to computations [14].

Diana et al. developed a product recommendation system that used an autoencoder based on a collaborative filtering technique. Collaborative filtering does not operate effectively if it was not presented with a sufficient quantity of data or if there were users with extremely diverse tastes from the others, as this filtering was dependent on user similarity [15]. Triyanna et al. presented a novel similarity method known as User Profiles Correlation-based Similarity, which experimented with genres data as well as users profile data, such as gender, age, employment, and locations. The weight of the similarity of user's rating and user behaviour values for movie suggestions was determined using all user profile data. However, this UPCSim method took longer to run than the preceding algorithms [16].

Alwesh proposed a feature selection wrapper method that used K-Nearest Neighbor classification and was subjected to two modifications using the MBO algorithm. The first modification involves utilising an enhanced crossover operator to improve feature selection. The second part integrates the Lévy flight distribution into the MBO to improve convergence speed [17]. Chen et al. implemented a deep learning model to process user comments and generate a possible user rating for user recommendations. First, the system used sentiment analysis to create a feature vector as the input nodes. Next, the system implemented noise reduction in the data set to improve the classification of user ratings. Finally, a DBN and sentiment analysis achieved data learning for the recommendations [18].

3 Proposed Methodology

A movie recommendation system based on Facebook user profiles using the Monarch Butterfly Optimization algorithm and Deep Belief Network is the proposed work. Based on the content-based filtering recommendation system, this proposed model is developed. Fig. 2 represents the block diagram of the proposed model. The proposed model recommends the movies for the users using their Facebook profiles.

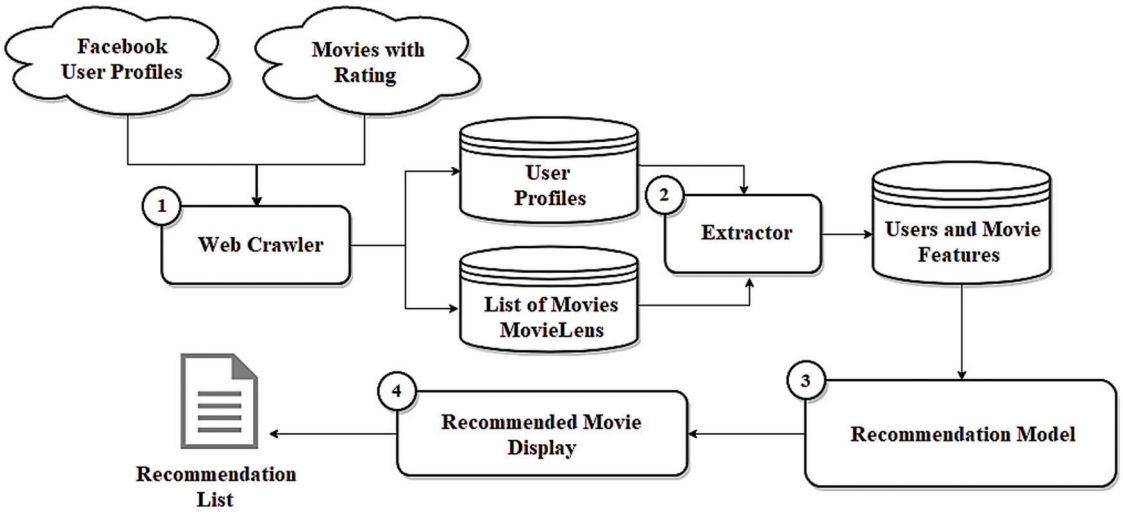

Figure 2: Work flow of proposed model

First, the system crawls the Internet, collecting Hypertext Markup Language (HTML) data from Facebook user-profiles and Movies with genres and ratings. The set of user-profiles, in particular, begins with the set of random seeds that link to the Uniform Resource Locators (URL) of profile. A dataset is created for each crawled profile. The retrieved URLs are then used to generate new URLs, and the processes are repeated unless there is no additional unknown user's profile to search. The tracking procedure is feasible because each profile links to various profiles via user links. The movie collection comprises a list of movies provided by the online movie database (i.e., Internet Movie Database (IMDb)). As a result, every HTML page connected to the movie URLs in the list was gathered and saved in the data collection.

Secondly, the crawled data saved in the datasets is filtered by the model. During the preprocessing step, it normalizes the pages of HTML's text coding and eliminates duplicate and noise data. The retrieved text features were then saved in the feature dataset. Finally, the model uses the stored features to generate vector representations of items and users. In the Vector Creation phase, the model extract features of the user and movie to generate word vectors that represent them. Each user and movie, in particular, is indicated by the vector of words taken from its text content. The model then changes the word vector in the Vector Filtering phase, stemming, removal of stop words, and removing accents, punctuations, and special characters to improve the feature of user and movie specifications. The model generates weight vectors for user and movie representations during the Vector Weighting phase. Remember that the frequency of the term and the specificity of the term scheme are used to compute the weights for each user and movie in the dataset to create the vectors User (u) and Content (i). The computing of similarities between users and movies is accomplished in the Similarity Computation phase, as specified in Eq. (1).

$$sim(u,i) = \frac{\vec{w}_u \cdot \vec{w}_i}{||\vec{w}_u||||\vec{w}_i||} = \frac{\sum_{x=1}^{y} w_{x,u} \times w_{x,i}}{\sqrt{\sum_{x=1}^{y} w_{x,u}^2} \times \sqrt{\sum_{x=1}^{y} w_{x,i}^2}} \tag{1}$$

Consider the Content (i), which is the text contents of item i expressed as the vector of weights. Because User (u) and Content (i) were both indicated as vectors $\vec{w}_u$ and $\vec{w}_i$. The cosine similarity function was used to calculate the sim (u, i) similarity among item i and user u. The value of similarity was produced by calculating the angle cosine among the two vectors, as shown in Eq. (1). As a result, the similarity value ranges from 0 to 1. The closer the vectors are to 0, the less equivalent they are, the closer the vectors are to 1, and the more similar are the vectors.

3.1 Monarch Butterfly Optimization

The Monarch Butterfly Optimization algorithm is a population-based method that falls under the class of swarm intelligences techniques, which were stimulated by the behaviour of swarming species such as bees and butterflies. These butterflies' migratory behaviour is mimicked in order to address different optimization issues. To get the optimal answer to the problem, certain guidelines and essential principles must be followed.

- All of the butterflies in the population may be found in either land 1 (home before migrations) or land 2 (after migrations).
- Every butterfly's child was created by the migration operators, despite whether the parents were presented in land 1 or *l* and 2.
- Because the populations must not vary and must always be stable, one of the two (either parent or new child) would be eliminated using the fitness function.
- The butterflies that were chosen on the basis of fitness function were moved on to the following generations and were not altered by the migrating operators [17].

3.1.1 Migration Operator

The butterfly migrating process was stated as,

$$X_{i.k}^{t+1} = X_{r1.k}^{t} \tag{2}$$

Where $X_{i.k}^{t+1}$ denotes the k^{th} element of X_i at t+1 generations, which indicates the position of butterfly i, and $X_{r1.k}^{t}$ denotes the k^{th} element of the location of next generation. In this case, r was the random integer generated using the equation,

$$r = rand * peri \tag{3}$$

Where *peri* denotes the migration period duration, if $r > p$, the k^{th} element of the location of next-generation was computed using the equation,

$$X_{i.k}^{t+1} = X_{r2.k}^{t} \tag{4}$$

where $X_{r2.k}^{t}$ denotes the k^{th} element of $Xr2$ at butterfly $r2$ generation t. As a result, p indicates the monarch butterfly ratio in land 1. The algorithm for the Migration operator is as follows,

---

Step1: Initialize

Step2: for i = 1 to NP1 (for each monarch butterfly in Subpopulation 1), **do**

Step3: for k = 1 to D (each element in i^{th} monarch butterfly), **do**

Create a number randomly *rand* by uniform distributions;

$r = rand * peri$;

Step4: if $r \leq p$, **then** select a monarch butterfly randomly in Subpopulation 1 (say r1);

Create the k^{th} elements of the new generation location as Eq. (2);

Step5: else

Select a monarch butterfly randomly in Subpopulation 2 (say r2);

Create the k^{th} elements of the new generation location as Eq. (4);

end if

end for k

end for i

end

3.1.2 Butterfly Adjusting Operator

By changing the p-value ratio, the equilibrium between the migration directions from l and 1 to l and 2 is accomplished. The total number of monarch butterflies in both the lands reflects the total population, referred to as NP. If p is big, the number of butterflies chosen from l and 1 is higher than the number of butterflies chosen from l and 2, and conversely. If the produced *rand* was less than or equivalent to the p-value, the position of the butterflies is changed. The updated position of the butterfly is shown in the following equation.

$$X_{j.k}^{t+1} = X_{best.k}^{t} \tag{5}$$

where $X_{j.k}^{t+1}$ indicates the k^{th} element of X_j at the t+1 generations that expressed the location of butterfly j, and $X_{best.k}^{t}$ indicates the k^{th} element of *Xbest* in both land 1 and land 2. If $rand > p$, the following equation is used to update it.

$$X_{j.k}^{t+1} = X_{r3.k}^{t} \tag{6}$$

If *rand* is larger than the butterfly adjustment rate (BAR), a new location was updated using the expression as follows,

$$X_{j.k}^{t+1} = X_{j.k}^{t+1} + \alpha * (dx_k - 0.5) \tag{7}$$

Where BAR denotes the butterfly's adjustment rate, and dx was the j's walk step determined by conducting Levy flight as follows,

$$dx = Levy\left(X_j^t\right) \tag{8}$$

α in Eq. (7) indicates the weighted factors that may be computed using the condition below,

$$\alpha = S_{max}/t^2 \tag{9}$$

where S_{max} is the maximum length of the one-step stride of butterfly and t is the present generation. The algorithm for this is as follows, [19]

Step1: Initialize

Step2: for i = 1 to NP2 (each monarch butterfly in Subpopulation 2), **do**

Compute walk step dx by Eq. (8);

Compute weighting factor by Eq. (9);

Step3: for k = 1 to D (each element in j^{th} monarch butterfly), **do**

Generate the number randomly *rand* by uniform distributions;

Step4: if $r \leq p$ **then**

Create k^{th} elements of the new generation location as Eq. (5);

Step5: else

Select the monarch butterfly randomly in Subpopulation 2 (say r3);

Generate k^{th} element of the new generation location as Eq. (6);

Step6: if $rand > BAR$, **then**

$$X_{j.k}^{t+1} = X_{j.k}^{t+1} + \alpha * (dx_k - 0.5)$$

end if

end if

end for k

end for j

end

3.2 Deep Belief Network

Many Restricted Boltzmann Machines (RBM) are layered on the DBN. The RBM theory is introduced first in this part, followed by the DBN building and training methods. The RBM is a generative stochastic artificial neural network that can match the input data's probability distribution. It is divided into two parts: input layer made up of neurons with input data and a hidden layer made up of neurons with the input layer's output. Some edges link neurons in the input and hidden layers, whereas neurons in the same layer are limited. Simultaneously, the two layers were symmetrically and bidirectionally linked [18]. Fig. 3 represents the RBM, where v indicates the input layer and h indicates the hidden layer. An energy function can be used to produce the probability distributions learnt by the RBM among the input and hidden layers. The following is an equation of the energy function.

$$E(v,h) = -h^T Wv - a^T v - b^T h = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{ij} W_{ij} v_i h_j \tag{10}$$

where v_i is the i^{th} input layer unit and h_j is the j^{th} hidden layer unit; W_{ij} was the connection weight among v_i and h_j; a_i and b_j were unit threshold values for v_i and h_j, respectively. Hence the joint probability distribution could be defined as shown below.

$$p(v,h) = \exp(-E(v,h)) \Big/ \left\{ \sum_{v,h} \exp(-E(v,h)) \right\} \tag{11}$$

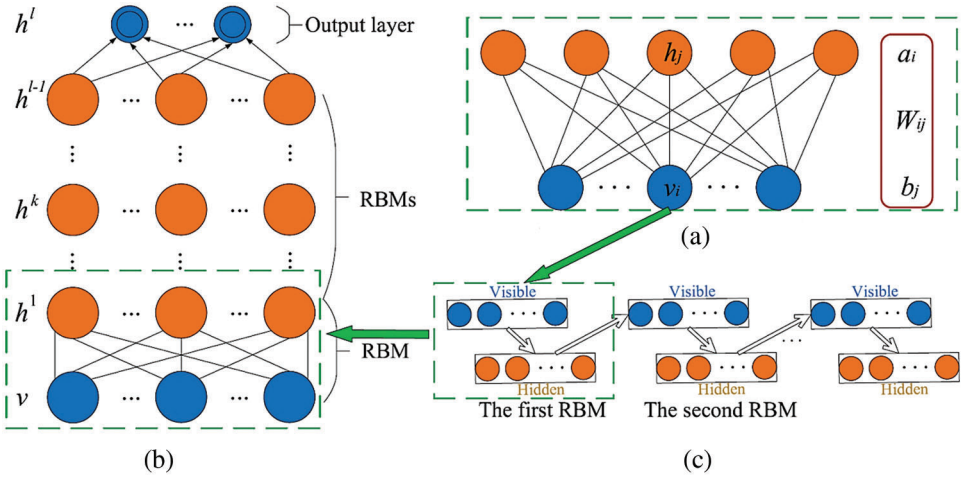

Figure 3: DBN implementation: (a) RBM; (b) Stacked procedures of RBM; (c) DBN [20]

As shown in Fig. 3, the probability of activation units in hidden layer is tentatively independent of the input layer's state. In most analyses, binary units were employed, where v_i and $h_j \in (0, 1)$. Consequently, the initial j hidden layer's probability nodes could be calculated as follows:

$$P(h_j = 1|v) = sigmoid\left(b_j + \sum_j W_{ij} h_j\right) \tag{12}$$

It can be provided for the entire hidden layer by:

$$P(h|v) = \prod_j (h_j|v) \tag{13}$$

The activation probability of nodes in the input layer is:

$$P(v_i = 1|h) = sigmoid\left(a_j + \sum_i W_{ij} v_i\right) \tag{14}$$

For the input layer, it was expressed as:

$$P(v|h) = \prod_i (v_i|h) \tag{15}$$

The probability $P(v)$ was the input vector v's probability across the hidden units, which was computed as follows:

$$P(v) = \sum_h \left\{ \exp(-E(v,h)) \Big/ \sum_{v,h} \exp(-E(v,h)) \right\} \tag{16}$$

As a result, the function of objective may be described as follows:

$$L(\theta, v) = \sum_{v \in S} \log P(v, \theta) \tag{17}$$

where $\theta \in \{a, b, W\}$; S indicates the dataset used for training.

The above-described optimal issue is based on Bayesian statistic theory and aims to answer the model's parameters by maximizing log-probability. To find a better solution to this issue, Hinton contrast divergence (CD), a fast-learning technique was used. The following two methods can support to speed up the sample procedure. One method is used as training sample for completing the Markov chain's initialization, while other is to acquire samples after Gibbs's sampling k-steps. It is known as CD-k. Several earlier experiments have demonstrated that CD performs well though $k = 1$ [20].

CD-1 was used in this work, and the updated rules for elements a, b, W in the RBM could be deduced from Eqs. (18)–(20), which are provided by:

$$W^{t+1} = W^t + \epsilon(P(h|v^{(0)})[v^{(0)}]^T - P(h|v^{(1)})\left[v^{(1)}\right]^T\Big) \tag{18}$$

$$a^{t+1} = a^t + \epsilon\left(v^{(0)} - v^{(1)}\right) \tag{19}$$

$$b^{t+1} = b^t + \epsilon\left(P\left(h|v^{(0)}\right) - P\left(h|v^{(1)}\right)\right) \tag{20}$$

here t denotes time steps and ε was the rate of learning.

Step1: Initialization

Input: $v^{(0)}$is a sample for training drawn from the RBM's training distributions;

Output: a, b, W are the updated parameters in the RBM

Step2: for each hidden unit j do:

calculate $P\left(h_j^{(0)} = 1|v^{(0)}\right)$ by using Eq. (12);

sample $h_j^{(0)} \in 0,\ 1$ from $P\left(h_j^{(0)} = 1|v^{(0)}\right)$;

end for

Step3: for each input unit i, do:

calculate $P\left(v_i^{(1)} = 1|h^{(0)}\right)$ using Eq. (14);

sample $v_i^{(1)} \in 0,\ 1$ from $P\left(v_i^{(1)} = 1|h^{(0)}\right)$;

end for

Step4: for each hidden unit j do:

calculate $P\left(h_j^{(1)} = 1|v^{(1)}\right)$ using Eq. (12);

sample $h_j^{(1)} \in 0,\ 1$ from $P\left(h_j^{(1)} = 1|v^{(1)}\right)$;

end for

$W = W + \epsilon\left(P\left(h|v^{(0)}\right)\left[v^{(0)}\right]^T - P\left(h|v^{(1)}\right)\left[v^{(1)}\right]^T\right)$;

$a = a + \epsilon\left(v^{(0)} - v^{(1)}\right)$;

$b = b + \epsilon\left(P\left(h|v^{(0)}\right) - P\left(h|v^{(1)}\right)\right)$;

Step5: return a, b, W;

As shown in Fig. 3, the DBN was a neural network with l layers. The initial layer was the input layer with inputs of vector x. The output layer h^l was the last layer. Hidden layers were those that exist between them and were symbolized by h^1, …, h^{l-1}. The DBN's hidden layers were built up of RBMs with sigmoid functions. The function of top activation would utilize the regression functions to forecast data flow. Its purpose was to learn the distribution of joint probability as shown in Eq. (21) across the vector x and the hidden unit.

$$P\left(x,\ h^1, \ldots, h^l\right) = \left(\prod_{i=1}^{l-1} P\left(h^{i-1}|h^i\right)\right) P\left(h^{l-1}, h^l\right) \tag{21}$$

The classification processes are then provided. The top layer's features are abstracted from the bottom layer's features. As a result, as differentiated to raw data, these features were significantly more significant and suited for inferences. Back propagation (BP) has conventionally been used to build neural networks with several layers, although it was prone to falling into local minima. The DBN goes deeper as more parameters were added, and the optimization issue becomes more complicated. One approach to overcoming the BP method's local minima was to assign some best values to the parameters. As a result, the deep network training process was separated into two stages: pretraining and fine-tuning. In the fine-tuning stage, the loss function was created, and the BP technique was used to minimally change the parameters of the DBN from top to bottom. Multiple RBMs are stacked to form the proposed DBN. The three phases in this approach are as follows. In pretraining, the initial independent RBM was pre-trained as a code utilizing the built training data. Second, as illustrated in Fig. 3, after obtaining the parameters in the initial

RBM, the initial RBM's output layer was used as the next RBM's input. This method was then reiterated up to the training was completed. Following the initial phase, the BP method was used to minimally change the parameter using the supervised pattern in order for the DBN's objective functions to reach the minima. The BP method goes through two major cycles: propagation and weight updating. During the procedure, all parameters are changed to generate an error. The error is then backpropagated in order to change the network settings and appropriately reduce the error.

4 Experiment Analysis

The experiments are carried out on a system with Intel(R)W-2223 CPU @ 3.9 GHz, 12 GB RAM, and NVIDIA's GTX1080Ti GPU platform. The OS utilized was Ubuntu 16.04, and the application was written in Python; the particular version of the software was 3.5. Tensor-flow 1.4.0 was utilized for implementing the deep learning module since the proposed model was based on the theory of deep learning.

4.1 Facebook Dataset

The dataset with user profile was collected from the Facebook Platform. This dataset was gathered from a real Facebook account established using the Facebook Application Programming Interface (API), which includes 565 friends and 2043 pairs of friends. The connection is low, and each user has four to five friends on average. The demographic data of the user is taken from the Facebook platform. User names, birthdays, gender, friend count, educational backgrounds, favourite artists, favourite movies, and language spoken are among the features gathered. The description of features collected from the dataset was shown in Tab. 1.

Table 1: Description of features

Features	
Gender	382-Male; 183-Female
Age	18–65
Education	High school-71; College-85
Work position	45 different categories
Favourite artists	93 different artists
Favourite movies	865 different movies
Languages	10 different languages

4.2 MovieLens 100K Dataset

MovieLens is a movie rating dataset collected through the ongoing MovieLens project. It is distributed by GroupLens Research at the University of Minnesota. Data was collected through the MovieLens website, which is publically available. MovieLens offers 100,000 ratings ranging from 1 to 5 stars for 1682 movies submitted by 943 individuals, with each user rating at least 20 films. The ratings range from 1 (poor) to 5 (outstanding). There are 19 movie genres (i.e., one category for each genre), and each film is assigned to at least one of them. This dataset can be downloaded from the Grouplens website (https://grouplens.org/datasets/movielens/100k/) [15].

The dataset (refer to Tab. 2) used in the experiment analysis is split into two segments. The first segment is called a practical dataset, and it contains 80 percent of data used for experiment training; the second segment is called a test dataset, and it contains 20 percent of data used for experiment testing. The Facebook and MovieLens features are evaluated for performance research to validate if the Facebook and

MovieLens data with various features can solve the matching recommendations. All of the files are compared to features to validate that the features will support movie recommendations. Recommending a few movies may lower the probability of users discovering a movie, but recommending too many movies may appear random and make it difficult for users to choose a movie.

Table 2: Description of MovieLens dataset

MovieLens dataset files	Features description
User	The user file includes demographic data about the 943 users. "User-ID | age | gender | occupation | zip code"
Item	The item file has data about the movies. "Movie ID | movie title | release date | video release date | IMDb URL | unknown | Action | Adventure | Animation | Children's | Comedy | Crime | Documentary | Drama | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller | War | Western"
Data	The data file comprises 100,000 ratings by 943 users on 1682 movies. "User ID | item id | rating | timestamp."

The MAE is a statistical measuring parameter used to assess the quality of recommendations [21–25]. If the MAE number is low, the algorithm recommends accurate recommendations. On test users, MAE estimates the divergence between the user profile and the movie, as indicated in Eq. (22). This parameter evaluation will provide high-quality recommendations.

$$MAE = \frac{\sum \left| P_{u,i} - r_{u,i} \right|}{N} \tag{22}$$

where $P_{u,i}$ represents the predicted recommendation for user u on movie i, $r_{u,i}$ represents the actual recommendation, and N indicates the number of total recommended movies.

The RMSE was the statistical accuracy metric utilized in this model. This gives greater weight to higher absolute errors.

$$RMSE = \sqrt{\frac{1}{N} \sum_{u,i} \left(P_{u,i} - r_{u,i} \right)^2} \tag{23}$$

When recommending movies, the smaller the RMSE, the more accurate the recommendation system will be.

For the evaluation, decision support accuracy criteria such as precision and recall are used. The proportion of recommended movies that are relevant to the user is used to calculate precision. This metric shows the number of recommended movies of interest to the user as the percentage of the number of total movies recommended to them.

$$Precision = \frac{TP}{TP + FP} \tag{24}$$

The recall was described as the proportion of similar movies that are also on the recommended list. Thus, it indicates the number of movies that were recommended to the user based on their interests.

$$Recall = \frac{TP}{TP + FN} \tag{25}$$

The recommended movies for the users generated from the proposed recommendation model is presented in Tab. 3. Based on the user's interest in the genre, the recommendation model recommends these movies. The performance evaluation of the proposed model is computed based on the MAE, RMSE, Precision, and Recall parameters.

Table 3: Movies recommended for the users

Users	Movie list (Recommended)
UID-1	MovID 1, 50, 260, 294, 348
UID-2	MovID 118, 174, 258, 300, 315
UID-3	MovID 44, 69, 108, 246, 365
UID-4	MovID 50, 88, 102, 127, 181

Tab. 4 represents the performance analysis of the proposed model based on these parameters. Figs. 4 and 5 represent the graphical plot of the performances analysis. The performances analysis of the proposed model was compared with the other existing models used in the recommendation systems for validation. FCM-BAT, CF-kNN+NDCG, UPCSim, and Deep Autoencoder models are the existing approaches used for the comparison with the proposed model. The proposed model outperforms each model compared with in terms of MAE, RMSE, Precision and Recall.

Table 4: Performance analysis of the proposed model

Users	MAE	RMSE	Precision	Recall
UID-1	0.732	0.913	93.21	92.85
UID-2	0.719	0.916	94.16	93.48
UID-3	0.725	0.919	95.33	94.82
UID-4	0.730	0.911	94.08	93.60

Tab. 5 represents the comparison of the performances analysis of the model with other existing models. The MAE and RMSE were the evaluation parameters for measuring the performances among the algorithms. The MAE and RMSE values of the proposed model performed on the datasets were 0.716 and 0.915, individually. Compared with the precision and recall, the proposed model achieved 97.35% and 96.60%, which are better than the compared models. Figs. 6 and 7 represent the graphical plot of the performance analysis comparison. As a result, the proposed model has the best recommendation performance in relation to MAE, RMSE, Precision and Recall compared with the existing algorithms discussed above.

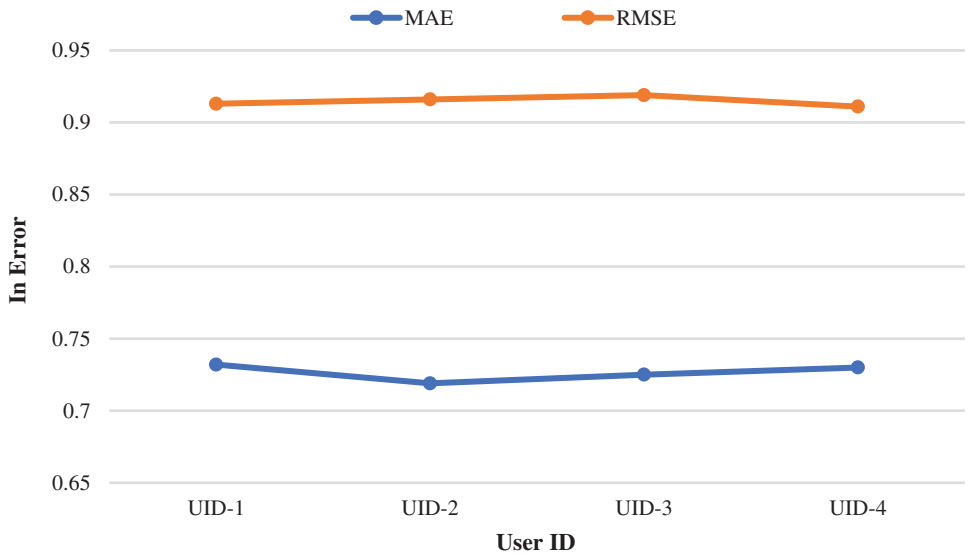

Figure 4: Graphical plot for proposed model's MAE and RMSE

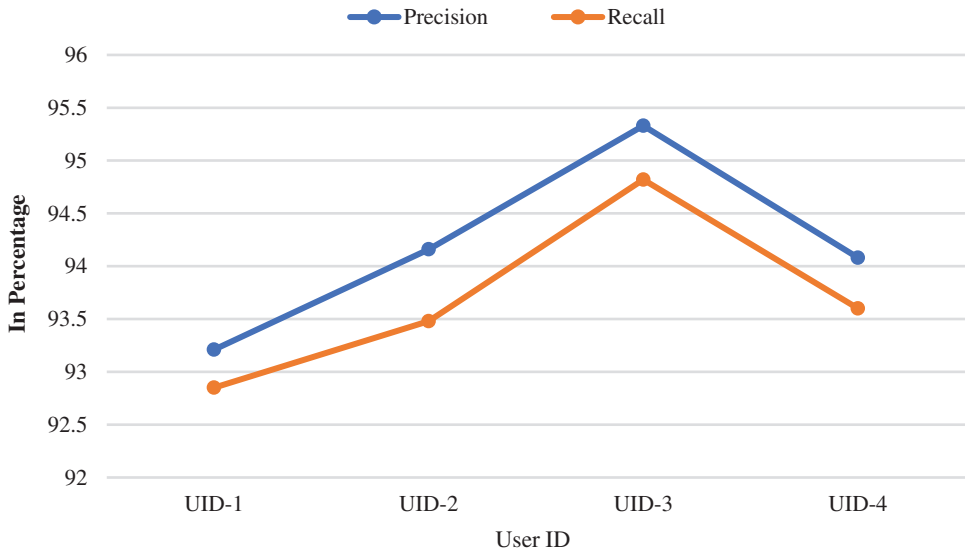

Figure 5: Graphical plot for proposed model's precision and recall

Table 5: Comparison of performance analysis

Models	MAE	RMSE	Precision	Recall
FCM-BAT	0.788	0.972	90.14	89.55
CF-kNN+NDCG	0.748	0.965	92.78	90.92
UPCSim	0.739	0.949	95.51	94.08
Deep Autoencoder	0.725	0.924	96.44	95.79
Proposed	0.716	0.915	97.35	96.60

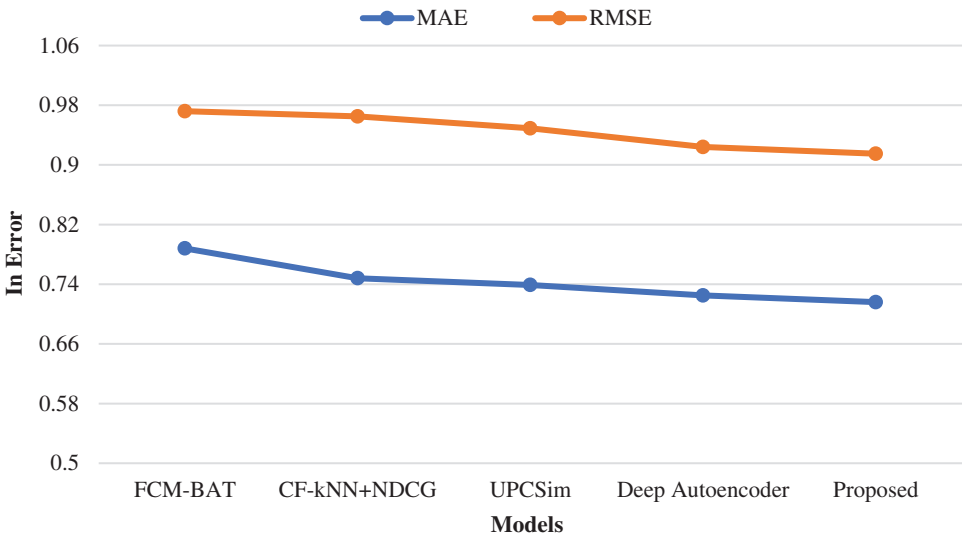

Figure 6: Graphical plot of MAE and RMSE comparison

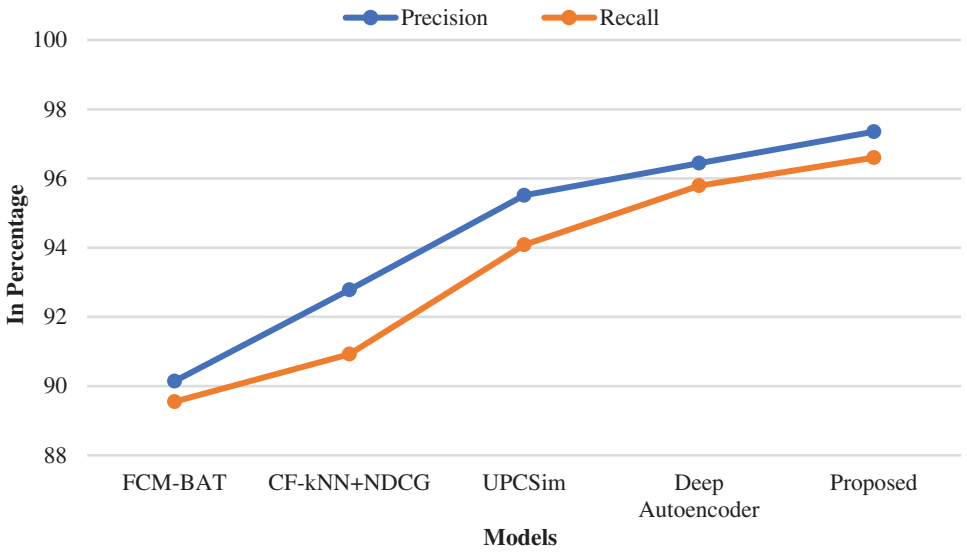

Figure 7: Graphical plot of precision and recall comparison

5 Conclusion

A movie recommendation model based on a content-based filtering approach using a hybrid deep learning model has been proposed in this research. The primary objective of this work was to develop a movie recommendation system that recommends movies based on the user's Facebook profile. The proposed model is built with a hybrid model that combines the Monarch Butterfly Optimization with the Deep Belief Network. For feature extraction, vector creation, filtering, and weighting were used, while feature selection used the Monarch Butterfly Optimization, and for classification, DBN was used. For performance evaluation, the Facebook dataset and the MovieLens dataset were used in this work. The dataset utilized in the experimentation was split into two segments, 80% used for training and 20% used for testing the experiment. The Facebook and MovieLens features are evaluated for performance research to validate if the Facebook and MovieLens data with various features can solve the matching

recommendations. All these files were compared to features to validate if they could allow movie recommendations. The proposed model's MAE and RMSE values were 0.716 and 0.915, and its precision and recall are 97.35% and 96.60%, respectively. Extensive tests have demonstrated the advantages of the proposed model in terms of MAE, RMSE, Precision, and Recall compared to state-of-the-art algorithms such as FCM-BAT, CF-kNN+NDCG, UPCSim, and Deep Autoencoder. In future, based on this proposed model, a job recommendation model can be developed by combining the LinkedIn profiles with the Job search platforms like Indeed, Naukri, etc., which could be helpful for the users to search for suitable jobs.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] G. Mahesh and C. Neha, "A review of movie recommendations system: Limitation, survey and challenges," *Electronic Letters on Computer Vision and Image Analysis*, vol. 19, no. 3, pp. 18–37, 2020.

[2] F. Zeshan, E. Mahsa, N. Dina, I. Ahmed and K. Rasha, "Recommendation system: Algorithm, challenges, metrics, and business opportunities," *Applied Sciences*, vol. 10, pp. 7748, 2020.

[3] V. J. Minal and U. B. Sneha, "Survey on movie recommendation approaches based on user preferences," *International Journal of Advanced Science and Research*, vol. 6, no. 4, pp. 118–121, 2021.

[4] Y. Wang, Y. Zhu, Z. Zhang, H. Liu and P. Guo, "Design of hybrid recommendation algorithm in online shopping system," *Journal of New Media*, vol. 3, no. 4, pp. 119–128, 2021.

[5] S. S. Ravi, A. S. Aijaz and L. Eldon, "Designing recommendation or suggestion systems: Looking to the future," *Electronic Markets*, vol. 31, no. 2, pp. 243–252, 2021.

[6] D. Nabanita, B. Surekha, D. Nilanjan and B. Samarjeet, "Social networking in web-based movie recommendations systems," in *Social Network Science: Designs, Implementations, Security, and Challenges*, In: N. Dey (Ed.), Cham: Springer, pp. 25–45, 2018.

[7] K. T. Ashish, M. Himanshu, S. Pranav and G. Siddharth, "A new recommendations system using maps-reduce-based tournament empowered Whale optimization algorithms," *Complex & Intelligent Systems*, vol. 7, no. 1, pp. 297–309, 2021.

[8] V. Subramaniyaswamy, R. Logesh, M. Chandrashekhar, A. Challa and V. Vijayakumar, "A personalised movie recommendations system based on collaborative filtering," *International Journal of High-Performance Computing and Networking*, vol. 10, no. 1/2, pp. 54–63, 2017.

[9] B. Ramzan, I. S. Bajwa, N. Jamil, R. U. Amin, S. Ramzan *et al.,* "An intelligent data analysis for recommendations systems using machine learning," *Scientific Programming*, vol. 2019, no. 5941096, pp. 1–20, 2019.

[10] S. R. S. Reddy, N. Sravani, K. Subramanyam, S. Ashok and B. Venkatesh, "Content-based movie recommendation system using genre correlation," in *Smart Intelligent Computing and Applications, Smart Innovation, Systems and Technologies*, In: S. C. Satapathy (Ed.), vol. 105, Singapore: Springer, pp. 391–397, 2019.

[11] V. Subramaniyaswamy, R. Logesh, D. Malathi, V. Vijayakumar, H. R. Karimi *et al.,* "Effective user preferences mining-based personalized movie recommendations systems," *International Journal of Computer Aided Engineering and Technology*, vol. 13, no. 3, pp. 371–387, 2020.

[12] V. Phonexay, X. Khamphaphone and P. Doo-Soon, "Movie recommendation system based on users' personal information and movies rated using the method of k-clique and normalized discounted cumulative gain," *Journal of Information Processing Systems*, vol. 16, no. 2, pp. 494–507, 2020.

[13] Y. Zhenning, H. L. Jong and Z. Sai, "Optimization of the hybrid movies recommendation systems based on weighted classifications and users collaborative filtering algorithms," *Complexity*, vol. 2021, no. 4476560, pp. 1–13, 2021.

[14] D. Debajit, T. M. Navamani and D. Rajvardhan, "Products and movie recommendation system for social networking sites," *International Journal of Scientific & Technology Research*, vol. 9, no. 10, pp. 262–270, 2020.

[15] F. Diana, S. Sofia, A. António and M. José, "Recommendation system using autoencoders," *Applied Sciences*, vol. 10, no. 16, pp. 5510, 2020.

[16] W. Triyanna, H. Indriana and B. A. Teguh, "User profiles correlation-based similarity (UPCSim) algorithm in movies recommendation systems," *Journal of Big Data*, vol. 8, no. 52, pp. 1–21, 2021.

[17] M. Alwesh, "Solving feature selection problems by combining mutation and crossover operations with the monarch butterfly optimization algorithm," *Applied Intelligence*, vol. 51, no. 6, pp. 4058–4081, 2021.

[18] R. Chen and Hendry, "User rating classification via deep belief network learning and sentiment analysis," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 3, pp. 535–546, 2019.

[19] A. Mohammed, S. Al Khalaileh, B. B. Gupta, A. Almomani, A. I. Hammouri *et al.,* "The monarch butterfly optimization algorithm for solving features selection problem," *Neural Computing and Applications*, vol. 23, no. 1, pp. 1–15, 2020.

[20] L. Linchao, L. Qin, X. Qu, J. Zhang, Y. Wang *et al.,* "Day-ahead traffics flow forecasting based on deep belief networks optimized by the multi-objective particles swarm algorithms," *Knowledge-Based Systems*, vol. 172, no. 1, pp. 1–14, 2019.

[21] E. Sreenivas and J. Praveena, "Performance analysis of deep belief neural network for brain tumor classification," *Journal of Computational Science and Intelligent Technologies*, vol. 1, no. 3, pp. 29–34, 2020.

[22] P. M. Surendra and S. Manimurugan, "A new modified recurrent extreme learning with PSO machine based on feature fusion with CNN deep features for breast cancer detection," *Journal of Computational Science and Intelligent Technologies*, vol. 1, no. 3, pp. 15–21, 2020.

[23] C. Narmatha, A. Hayam and A. H. Qasem, "An analysis of deep learning techniques in neuroimaging," *Journal of Computational Science and Intelligent Technologies*, vol. 2, no. 1, pp. 7–13, 2021.

[24] N. N. Anjum, "A study on segmenting brain tumor MRI Images," *Journal of Computational Science and Intelligent Technologies*, vol. 2, no. 1, pp. 01–06, 2021.

[25] R. Khilar, K. Mariyappan, M. S. Christo, J. Amutharaj, T. Anitha *et al.,* "Artificial Intelligence-based security protocols to resist attacks in Internet of Things," *Wireless Communications and Mobile Computing*, vol. 2022, no. 1440538, pp. 1–10, 2022.