


A Deep Autoencoder-Based Hybrid Recommender System

Yahya Bougteb, LM2I Laboratory, ENSAM, Moulay Ismail University, Morocco*

Brahim Ouhbi, LM2I Laboratory, ENSAM, Moulay Ismail University, Morocco

Bouchra Frikh, École Nationale des Sciences Appliquées de Fès, Morocco

Elmoukhtar Zemmouri, ENSAM, Moulay Ismail University, Morocco

 <https://orcid.org/0000-0001-9165-0858>

ABSTRACT

Recommender systems build their suggestions on rating data, given explicitly or implicitly by users on items. These ratings create a huge sparse user-item rating matrix, which opens two challenges for researchers on the field. The first challenge is the sparsity of the rating matrix, and the second one is the scalability of the data. This article proposes a hybrid recommender system based on deep autoencoder to learn the user interests and reconstruct the missing ratings. Then, SVD++ decomposition is employed, in parallel, to hold information of correlation between different features factors. Additionally, the authors gave a deep analysis of the top-N recommender list from the user's perspective to ensure that the proposed model can be used for practical application. Experiments showed that the proposed model performed better with high-dimensional datasets and outperformed various hybrid algorithms in terms of RMSE, MAE, and in terms of the final top-N recommendation list.

KEYWORDS

Deep Autoencoder, Deep Learning, Factorization Machines, Learning User's Profile, Recommender Systems, Sparsity Problem, SVD++, Top-N Recommendation List

1. INTRODUCTION

Retaining satisfied and happy users is a big challenge for Recommender System (RS). RS can help users find things of interest that might not be expected or have seen before. For example, a movie or a restaurant that matches user preferences but is not quite popular (Suriati et al., 2017). Recommender systems are usually classified into four models (Fakhfakh et al., 2017): content-based recommender system, Collaborative Filtering (CF), hybrid recommendation, and demographic filtering.

- **Content based recommender system:** This model is based on the comparison between items and users features. The RS recommends items which are similar to the ones that the user implicitly or explicitly interacts with previously through rating or clicking. Hence, a user profile is created and used to identify new interesting and relevant items for the user (Fakhfakh et al., 2016; 2017).

DOI: 10.4018/IJMCMC.297963

*Corresponding Author

This article published as an Open Access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

- **Collaborative filtering:** This approach uses the opinions of user's community, or the information about the past behavior, to predict the items that are interesting for a given user (Ouhbi et al., 2018). The similarity in taste of two users is calculated based on the similarity in the rating history of the users.
- **Hybrid recommendations:** This model combines two or more algorithms to create a new and better recommender system taking advantage of the strength and weakness of each one. The combination of the different techniques generates better and/or more precise recommendations.
- **Demographic filtering:** Unlike collaborative filtering, this model generates its recommendations based on user's demographic information. The RS classifies users under a set of demographic classes representing some demographic characteristics of users known from their nationality, age, gender, location etc.

Traditional recommender systems are insufficient when dealing with high-dimensional datasets and data sparsity. During the first decade, recommender systems have been seen to be effective and efficient in improving business performance. Hence, many algorithms were proposed to solve some drawbacks in real situations like in cold start problem situation (Wei et al., 2017; Wu et al., 2017; Hdioud et al., 2017).

Traditional CF algorithms have limited learning capacities and suffer from data sparsity problem. Recently, this problem was tackled using Cross Domain Collaborate Filtering (CDCF) which is a new way to alleviate the sparsity problem in the recommender systems (Yu et al., 2018; 2019). However, these methods cannot learn an effective high-order feature interactions between users and items, and it do not take into account implicit ratings which are in some sort an implicit interest in the item no matter what the rating was.

Recently, there are many attempts to develop more sophisticated models to solve sparsity and high dimensional problems. Deep learning has witnessed a great success in many application fields such as object recognition, speech recognition, computer vision and natural language processing. There is a tendency to use deep learning to address these problems (Ouhbi et al., 2018; Cheng et al., 2016; Guo et al., 2017; Kuchaiev & Ginsburg, 2017; Zhou et al., 2018; Song et al., 2019).

RNN and CNN were used to learn the interactive relationship between users and items (Wu et al., 2020). An attention mechanism was added into the RNN and CNN networks for better recommendation accuracy. The Hit Rate (HR) and the Normalized Discount Cumulative Gain (NDCG) were used as evaluation indicators for the top-n recommendation task. However, the authors mentioned that their paper mainly focused on off-line training which is inadequate for practical applications.

Deep Autoencoders are very powerful in reducing the dimension of data. They can learn an effective feature representation and capture the similarity between users and items (Vincent et al., 2010). Autoencoders were also used to apply the collaborative filtering intuition to neural networks (Sedhain et al., 2015). A simple autoencoder is a network which implements two transformations: Encoder and decoder: The encoder maps the input data into a low-dimensional encoding latent representation which is then mapped back to a reconstructed vector. Therefore, we propose a deep autoencoder to find the low-dimensional representation of the sparse user-item rating matrix while trying to learn the high-order feature interactions between users and items. First, we train the deep autoencoder on training datasets. Then, we reconstruct all the missing ratings. However, the correlation between the user-feature and item-feature factors in the dense representation could be lost. Hence, SVD++ decomposition is employed in parallel with deep autoencoder to extract extra user and item features for an effective learning, and a better prediction of the user interests. Traditional factorization methods like SVD or HOSVD (Higher-Order Singular Value Decomposition) handle data entries very poorly. For example, missing rating or unknown data are replaced by zeros. Our proposed method implements SVD++ because it provides a loss function (see equation 12) which does take into consideration the implicit interactions which are in some sort an implicit interest in the item regardless of the rating. Hence, SVD++ makes a good solution for unknown ratings (Koren, 2008).

Evaluating a recommendation system model by relying only on some metrics like RMSE and/or MAE, does not really show how the system will recommend items and how satisfied a user will be with the final top-n recommendation list. One solution to this problem is to use a generic re-ranking framework that customizes balance between accuracy and coverage (Zolaktaf et al., 2018). Another solution is the use of the transfer to rank algorithms such as holistic transfer to rank (HoToR), where authors converted the original feedback into three different but related views of examinations, scores, and purchases. These views enable addressing the uncertainty and the inconvenience challenges in the existing works (Ma et al., 2021). However, most of work done in this field using deep learning algorithms alone or in an ensemble approach with other algorithms does not take into account the final top-n recommendation list from users' view. Therefore, we made a deep analysis of the top-n recommendation list from a user perspective to ensure that our proposed model can be useful for practical application such as e-commerce (sell/buy items on a website for example) or video streaming applications.

In this paper, we give a state of the art to discuss the main proposed approaches. Moreover, we develop our hybrid model based on two components: a deep autoencoder and a SVD++ decomposition method. Deep autoencoder can generalize better to unseen feature combinations through low-dimensional dense embeddings learned from the sparse features, while SVD++ can hold the information of correlation between the user-feature and item-feature factors which can be defined as a Memorization of feature interactions (Guo et al., 2017). The benefits of generalization and memorization are combined in one hybrid model to solve sparsity, high dimensional, and implicit feedback problems.

The rest of this paper is organized as follows: Section 2 gives a short survey where we summarize the state-of-the-art of recommender systems methods with their datasets, advantages, and drawbacks. Section 3 provides the preliminaries of our research on the recommender systems as well as on the deep learning. Section 4 discusses in details the proposed method. Section 5 is devoted to the experiments and results. Finally, Section 6 presents the conclusion and future works.

2. RELATED WORKS

Inspired by the success of deep learning in natural language processing (Bahdanau et al., 2014) and computer vision (Huang et al., 2017), deep learning based methods have been proposed for recommendation systems. Deep learning discovers intricate structure in large datasets by using the back propagation algorithm to indicate how a machine should change and adapt its internal parameters that are used to compute the representation in each layer from the representation in the previous layer (LeCun et al., 2015).

Deep autoencoders are very powerful in terms of reducing the dimension and outperform Principal Component Analysis (PCA) in image compression (Côté & Larochelle, 2016). Deep autoencoders were used in topics detection (Bougteb et al., 2019; Rekik & Jamoussi, 2016) as well as in recommender systems (Strub et al., 2016; Kuchaiev et al., 2017; Sedhain et al., 2015). In (Wang & Wang, 2014), authors used the Hierarchical linear model with Deep Belief Network (HLDBN) based on a probabilistic graphical model and Deep Belief Network (DBN). They first obtain latent features for all songs by conducting matrix factorization, and then used DBN to map audio content to the latent features. According to the paper, DBN has seen best results, outperformed the traditional extracting method in the music field, and increased the effectiveness of the results.

In (Vincent et al., 2010), authors implement a Stacked Denoising Autoencoder (SDAE) to learn an effective feature representation and to capture the similarity between users and items. Then, collaborative filtering is used to make recommendation. Also, SDAE was used in a hybrid model combining deep learning method for feature extraction and collaborative filtering model into one framework called IRCD (Wei et al., 2017). Therefore, the hybrid model can predict rating of the complete cold start (CCS) and incomplete cold start (ICS). Authors trained the SDAE to learn useful

representations in a deep network using the descriptions of online retrieved items using OMDB API, while the timeSVD++ model was used to predict the unknown ratings (Wei et al., 2017).

To overcome the problem of data sparsity, authors in (Strub et al., 2016) proposed a hybrid recommender system based on autoencoders and performed collaborative filtering with side information by injecting that information to each layer input of the network instead of adding it to the first layer only. Thus, the dynamic autoencoders representation is forced to integrate this new data. Autoencoders were also used in (Sedhain et al., 2015) to apply the collaborative filtering intuition to neural networks.

A session-based recommender system is a system that analyzes the past behavior of users individually or a user community as a whole to detect patterns in the data and interest drifts of users over time (Quadrana et al., 2018; Moore et al., 2013; Fang et al., 2019). In (Hidasi et al., 2015), authors proposed an approach based on RNN for session-based recommendations to model the whole session. Their approach considers practical aspects of the task and introduces several modifications to the classic RNNs such as a ranking loss function. (Jing & Smola, 2017; Wu et al., 2017) include temporal information as features and supervision in their RNN based models. These features are concatenated with input, which provides limited benefit (Beutel et al., 2018).

Considering low and high order feature interactions simultaneously, (Cheng et al, 2016) combined the power of wide (generalized linear model) and deep (a feed-forward neural network) components to combine the benefits of memorization and generalization for recommender systems. Inspired by this hybrid approach, (Guo et al., 2017) proposed DeepFM, another ensemble approach that combines the power of factorization machines (wide part) and deep neural networks (deep part) to overcome the problem of the need for expertise feature engineering of the input.

(Zhou et al., 2018) used a Deep Interest Network which introduces a local activation unit to adaptively learn the representation of user interests from historical behaviors with respect to a given ad to point out the limit of using fixed-length vector to express user's diverse interests. Also, they developed two novel techniques to help training industrial deep networks: mini-batch aware regulator to save heavy computation of regularization and "Dice" a generalized PReLU (Parametric ReLU) to take into account that each layer follows different distribution.

(Song et al., 2019) proposed an algorithm based on the latest techniques in the literature of deep learning: attention and residual networks (Bahdanau et al., 2014; He et al., 2016). Attention was first used in the context of neural machine translation and has been proven to be effective in various tasks like question answering (Sukhbaatar et al., 2015) and recommender systems (Zhou et al., 2018). (Vaswani et al., 2017) proposed multi-head self-attention to model complicated dependencies between words in machine translation as researchers often ignore contextual information in their deep learning-based recommender systems like time, location, and interfaces.

(Beutel et al., 2018) bridged the contextual collaborative filtering literature and neural recommender literature to make use of contextual data in deep neural recommenders (particularly in RNN models). The authors demonstrated that prevailing techniques miss a significant amount of information in these features and offered "Latent Cross", an easy-to-use technique to incorporate contextual data in the RNN by embedding the context feature first and then performing an element-wise product of the context embedding with model's hidden states. They performed two sets of experiments: the first one was on a restricted dataset where time is the only contextual feature and the second one was on their production model (on YouTube). They got the best result for both Precision and MAP (Mean-Average-Precision). Table 1 briefly summarizes the most relevant approaches previously discussed.

3. PRELIMINARIES

Recommender systems depends on users explicit or implicit inputs to provide relevant suggestions. Rating based input create a huge sparse matrix which leads to a couple of challenges for researchers to deal with. The first challenge is the sparsity of the rating matrix and the second one is the scalability

Table 1. Topics detection methods summary

Article	Deep Learning Method	Approach	Metric	Dataset	Advantages	Weaknesses
(Hidasi et al., 2016)	GRU (RNN)	Session based CF	Recall MRR	RSC15 Collection from a Youtube like OTT video service	Whole session model provides more accurate recommendations	Complex model. No online evaluation. Insignificant Improvement.
(Cheng et al., 2016)	DNN	Hybrid (generalized linear model and DNN)	Offline AUC Online Acquisition Gain	Google play	Combined the wide and deep learning components. Best online A/B test result.	Depends on expertise feature engineering in the input of the wide part (the generalized linear model).
(Guo et al., 2017)	DNN	Hybrid (DeepFM)	AUC Logloss	Criteo Company dataset	Combines factorization machines and DNN to extract the lower-order and higher-order feature interactions. No expertise feature engineering dependencies.	The model has not been tested on high-dimensional datasets
(Kuchaiev & Ginsburg, 2017)	Deep autoencoder	Item-based (CF)	RMSE	Netflix	Fast training algorithm based on iterative output refeeding. Overcome natural sparseness of collaborate filtering.	Needs massive amount of data for training to give good RMSE.
(Zhou et al., 2018)	DIN	Session-based CF	AUC RelatImpr	Amazon (Electro) MovieLens Alibaba	Captures the diversity of user interests by introducing a local activation unit.	Does not take into account users' multiple concurrent interests in their historical behaviors.
(Song et al., 2019)	Self-Attentive ResNet	CF (AutoInt)	AUC Logloss	Criteo Avazu3 KDD124 MovieLens IM	Models explicitly different orders of feature combinations. Handle large-scale high-dimensional, sparse data.	Does not take into account the contextual information
(Beutel et al., 2018)	RNN	Hybrid	Precision MAP	YouTube	The model makes use of contextual data. Tested on a large-scale real-world production model and gave best results.	Does not tackle the cold start problem.
(Wu et al., 2020)	RNN & CNN	Hybrid	NDCG HR	MovieLens -100k	Adding an attention mechanism to the neural networks gave better accuracy	Requires long training time. Not applicable for some practical applications such as news recommendation.

of the data (large number of possible registered users and items). Autoencoders work better when trained on large volume of data and can solve these challenges. Autoencoder is able to reconstruct the original rating matrix based on a dense representation and able to learn high-order features.

Factorization methods find the latent features that might exist between users and items. These features cause individuals to rate items differently. For example, how dramatic a movie is? Or how funny a book is? These questions are form of latent features that factorization methods can find in a smaller dimension without even knowing what they really mean. SVD is a factorization method

that runs PCA on the users' matrix R and the items matrix R^T , then returns the matrices U and V^T respectively. U and V^T matrices are factors of the original user-item rating matrix R as shown in equation 1:

$$R_{(m \times n)} = U_{(m \times m)} \cdot S_{(m \times n)} V_{(n \times n)}^T \quad (1)$$

where S is an $m \times n$ rectangular diagonal matrix, m the number of users and n the number of items.

3.1 Autoencoder Model

An autoencoder is a network which implements encoder and decoder transformations to reconstruct the original matrix. The encoder is responsible for mapping the input $X' \in R^d$ to a low-dimensional encoding $z \in R^d$ ($d' < d$). A deep autoencoder is an autoencoder with more hidden layers. We used ReLU (Rectified Linear Unit) as activation function in the hidden layers (see equation 2 and figure 1), the sigmoid function in the output layer for ratings predictions, and Adam optimizer to minimize the error in the loss function (equation 4):

$$\begin{aligned} z &= f(WX + b) \\ &= \max(WX + b, 0) \end{aligned} \quad (2)$$

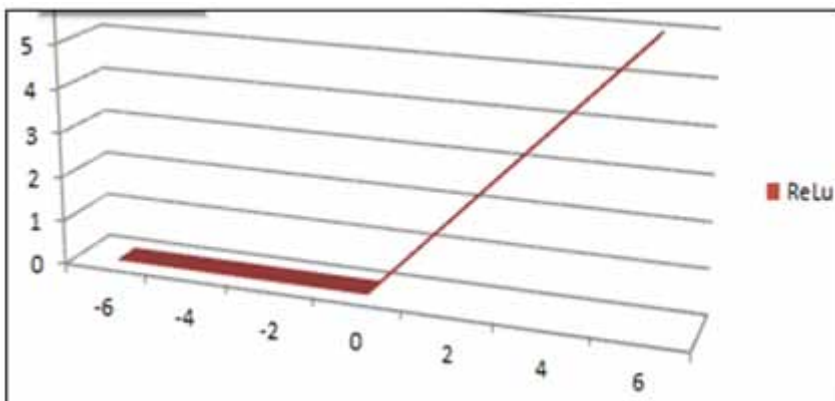
where W and b are the weight matrix and the encoding bias vector respectively. The latent representation z is then mapped back to a reconstructed vector $X' \in R^d$. The decoding transformation is performed using a separate decoding matrix:

$$X' = f(W^*z + b^*) = \max(W^*z + b^*, 0) \quad (3)$$

where W^* and b^* are the decoding matrix and the decoding bias vector respectively.

The autoencoder can find the low-dimensional representation of a matrix by minimizing the Mean Squared Error (MSE) or Mean Squared Deviation (MSD) between the input and the output layer (see equation 4):

Figure 1. ReLU activation Function



$$MSE(X, X') = \frac{1}{N} \sum_{i=1}^N (X_i - X'_i)^2 \quad (4)$$

3.2 Matrix Factorization

Matrix decomposition methods also known as matrix factorization uses both users and items matrices (R and R^T respectively) to hold the information of correlation between the user-feature and item-feature factors. Another strong point of matrix factorization is ability to incorporate additional information. When no explicit feedback is available, recommender systems can infer user preferences using implicit feedbacks by observing user behavior including browsing history, purchase history, clicking (Koren et al., 2009), etc. However, traditional factorization methods like SVD or HOSVD (Higher-Order Singular Value Decomposition) handle data entries very poorly. For example, missing rating or unknown data are replaced by zeros. Hence, we suggest using SVD++ instead of SVD.

In SVD we have to work with the loss function while running SGD (Stochastic Gradient Descent). The prediction \hat{r}_{ui} is set as:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u \quad (5)$$

where μ the mean of all ratings, and are the user bias and the item bias respectively, and are the item and the user factors respectively. For a given item i , measures the extent to which the item possesses those factors, positive or negative. For a given user u , measures the extent of interest the user has in items that are high on the corresponding factors (positive or negative). If user u is unknown, then the factors and the bias are set to zero. The same applies for item i with and Equation 5 is equivalent to Probabilistic Matrix Factorization when baselines are not used (Salakhutdinov & Mnih, 2008):

$$\hat{r}_{ui} = q_i^T p_u \quad (6)$$

To estimate all the unknown parameters, we minimize the regularized squared error:

$$\sum_{r_{ui} \in R_{train}} (r_{ui} - \hat{r}_{ui})^2 + (b_i^2 + b_u^2 + \|p_i\|^2 + \|q_u\|^2) \quad (7)$$

The minimization is performed by a straight forward SGD (Stochastic Gradient Descent):

$$b_i \leftarrow b_i + y(e_{ui} - \lambda b_i) \quad (8)$$

$$b_u \leftarrow b_u + y(e_{ui} - \lambda b_u) \quad (9)$$

$$q_i \leftarrow q_i + y(e_{ui} p_u - \lambda q_i) \quad (10)$$

$$p_u \leftarrow p_u + y(e_{ui} q_i - \lambda p_u) \quad (11)$$

where $e_{ui} = r_{ui} - \hat{r}_{ui}$. We perform these steps over all the ratings of the training set. Item and user factors are randomly initialized according to a normal distribution. Baselines are initialized to zero. The learning rate γ and the regularization term λ are initialized by default to the values 0.005 and

0.02 respectively. Then we try to find the optimal values during the training phase that minimize the MSE error and training time.

In SVD++, the loss function takes into account implicit ratings which are in some sort an implicit interest in the item no matter what the rating was. Thus, SVD++ makes a good solution for unknown (missing) ratings (Koren, 2008). The prediction \hat{r}_{ui} is defined as follows:

$$\hat{r}_{ui} = u + b_u + b_i + q_i^t \left(pu + |I_u|^{-1/2} \sum_{j \in I_u} y_j \right) \quad (12)$$

where the y_j terms are the item factors that capture implicit ratings. An implicit rating here describes the fact that a user u rated an item j , regardless of the rating value. If user u is unknown, then the bias b_u and the factors p_u are set to zero. The parameters are learned using the SGD on the regularized squared error function just like in SVD.

4. PROPOSED METHOD

The architecture of our deep autoencoder is illustrated in figure 2. Figure 3 shows the framework of our proposed model: It is an ensemble approach where individual models are trained separately. We combine both generated matrices in a weighted average using equation 2 which makes up the final predicted rating matrix. Finally, we calculate the RMSE and MAE, rank the results, and create a top-n recommendation list for a user u on an item i :

$$\hat{r}_{ui_{Hyp}} = \frac{\sum_{k=1} (w_k \cdot \hat{r}_{ui_{model\ k}})}{\sum_{k=1} w_k} \quad (13)$$

where w_k is the weight of the model k and $\sum_{k=1} w_k = 1$. Therefore, equation 13 becomes:

$$\hat{r}_{ui_{Hyp}} = \sum_{k=1} (w_k \cdot \hat{r}_{ui_{model\ k}}) \quad (14)$$

and in our case:

$$\hat{r}_{ui_{Hyp}} = w \cdot \hat{r}_{ui_{SVDpp}} + (1 - w) \cdot \hat{r}_{ui_{DAE5}} \quad (15)$$

where DAE5 is our deep autoencoder (with 5 hidden layers) and $w \in [0,1]$.

As illustrated in figure 3, a deep autoencoder was first proposed to find the low-dimensional representation of the sparse user-item rating matrix while in the same time is trying to learn the high-order feature interactions between users and items. First, we train it with all the data we have on training dataset. Then, we reconstruct the missing ratings with the trained model. However, we still have the problem of losing some information about the correlation between the user-feature and item-feature factors in the dense representation. Hence, we propose using SVD++ decomposition method in parallel with deep autoencoder model to keep correlation information for a better prediction and an effective learning of the user interests.

4.1 Activation Function

In order to choose the right activation function, in our case, we conducted some experiments by comparing ReLU (Rectified Linear Unit) with ELU (Exponential Linear Unit) performances on MovieLens 100K dataset. Table 2 shows the results.

According to the result in table 2, ReLU has slightly better RMSE and MAE results with deep networks (3 and 5 layers). We did not compare ReLU to other activation functions such as tanh and sigmoid for example because both ELU and ReLU outperformed most of the activation functions and they are the most popular choice in deep learning (Kuchaiev & Ginsburg, 2017). The advantages and disadvantages are listed in table 3.

4.2 Neurons

Finding the right number of neurons in each layer is very important in any deep learning algorithm. Therefore, we have tested different values for each layer to find the best number of neurons for our proposed model to minimize the mean squared error between the input and the output. The formula for calculating total number of nodes in hidden layers (Liu et al., 2019) is given in equation 16:

$$k = \sqrt{n + m} + t$$

where k is the number of nodes in the hidden layer, n denotes the number of nodes in the input layer, m indicates the number of nodes in the output layer ($n = m$ in an autoencoder), and t is a constant between [1, 10]. The architecture of our deep autoencoder is illustrated in figure 2.

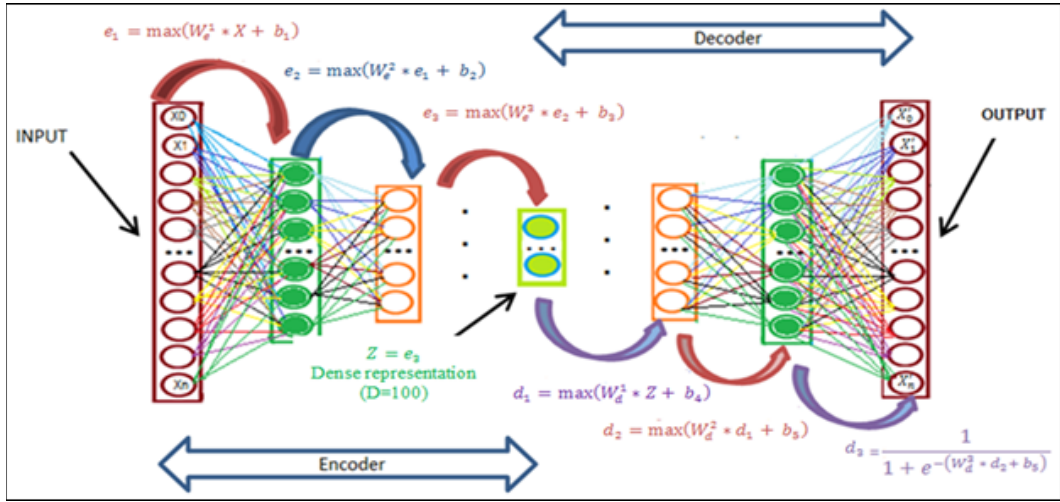
Table 2. Performance comparison between ReLU and ELU

	MAE			RMSE		
Hidden layers	1	3	5	1	3	5
ReLU	0.915	0.867	0.866	1.168	1.097	1.075
ELU	0.913	0.868	0.870	1.167	1.099	1.109

Table 3. Advantages and disadvantages of some activation functions

Activation Function	Advantages	Disadvantages
Tanh	The output is zero centered.	Vanishing Gradient problem, hard to train on small datasets.
Sigmoid	Easy to apply and train on small datasets (Srinivasan et al., 2019).	Vanishing Gradient Problem, the output is not zero centered.
ReLU	Solve vanishing gradient problems.	Hard to train on small datasets for learning non-linear behavior. Can only be used with a hidden layer (Srinivasan et al., 2019).
ELU	ELU becomes smooth slowly until its output equal to $-\alpha$. Unlike ReLU, ELU can produce negative outputs.	For $x > 0$, it can blow up the activation with the output range of $[0, +\infty]$

Figure 2. Our deep autoencoder architecture composed of 5 hidden layers: 3 for encoding the input and 3 for decoding



5. EXPERIMENTAL STUDIES

We evaluated our proposed method against two hybrid models: the first model was a DBN based model combined with CF (Ouhbi et al., 2018) and the second one was a deep autoencoder based hybrid model combined with ContentKNN (K-nearest neighbors) (Luong et al., 2012). We analyzed their performance on three well-known datasets: MovieLens 100K, MovieLens 1M, and Book-Crossing datasets.

5.1 Datasets Description

Book-Crossing Dataset was collected by Cai-Nicolas Ziegler in a 4-week crawl (August - September 2004) from the Book-Crossing community. This dataset contains 278,858 users (with demographic information) providing 1,149,780 ratings (explicit / implicit) about 271,379 books in a range scale of 1 to 10. Over 62% of all ratings are zeros which make the dataset extremely sparse. Therefore, we took only users that rated at least 10 different books on the training set which means we used only 12306 users.

Unlike Book-Crossing, MovieLens 100K and MovieLens 1M do not require any preliminary steps or clean up. MovieLens 100K is a stable benchmark dataset containing 100,000 ratings from 1000 users on 1700 movies. MovieLens 1M contains 1 million ratings from 6000 users on 4000 movies. Each user has rated at least 20 movies.

5.2 Experiments

Table 4 shows observations based on experiments were conducted to determine the amount and the effectiveness of hidden layers. It was observed that deep networks provide the best results. Note that after 5 hidden layers we noticed a small improvement regarding MAE while RMSE remains the same and the model started to over-fit because deep networks need massive amount of data for training. Therefore, we took DAE5 as part of our hybrid model.

To measure the accuracy of the predicted ratings we split our dataset into two sets. Three quarters of the dataset was devoted to the training set and the remaining quarter for the testing set. Then, we built a leave one out train/test split for evaluating the top-n recommendation list. However, we had to normalize all the ratings into the range $[0, 1]$ before feeding any data to the input of our deep autoencoder and restore them back to their original ranges after the training. We gave zero for every

Table 4. The effectiveness of hidden layers

	DAE1	DAE3	DAE5	DAE7
Number of hidden layers	1	3	5	7
MAE	1.130	1.092	0.857	0.856
RMSE	1.450	1.402	1.065	1.065

missing rating and we kept only users with more than 10 ratings for 10 different items ($n_{ratings} = 10$) on the Book-Crossing dataset. because of sparsity. Furthermore, we have tried different values for the number of ratings that we should take from this dataset ($n_{ratings} = 5, 10, 30$) and we took only the one that gave best results in terms of MSE and training time (10 in our case). Unlike Book-Crossing dataset, we used all users from MovieLens datasets.

Instead of replacing missing ratings with zeros, we could use CF to infer a missing entry (i, j) in the user-item rating matrix $R_{m \times n}$ with an entry (p, q) describing the ratings given by the p th user (first 10-nearest neighbor to the user i) to the q th item. However, learning these features interactions between users and items is a deep autoencoder task; it will denoise the input from zeros and predict all ratings by reconstructing the input. Furthermore, there is no user rating of 0.5 found in MovieLens 1M and only 1% of users on MovieLens 100K gave a rating of 0.5. Therefore, there would be no confusion between zero that represents a missing rating and a zero that represents a dislike (0.5 as rating). Thus, all zeros in the input matrix will be considered missing ratings.

We used each individual user in our training set as a set of inputs into our deep autoencoder. Therefore, we trained our model on users rather than items unlike what was done in (Sedhain et al., 2015). We fed in the ratings for every item for each given user in the input.

5.3 Results

We run experiments on MovieLens 100k with three different hybrid models: our proposed model, a deep autoencoder combined with ContentKNN in a hybrid model, and a Deep Belief Network (3 hidden layers) combined with CF using different values for the weights to find the optimal weights. The results are given in figure 4.

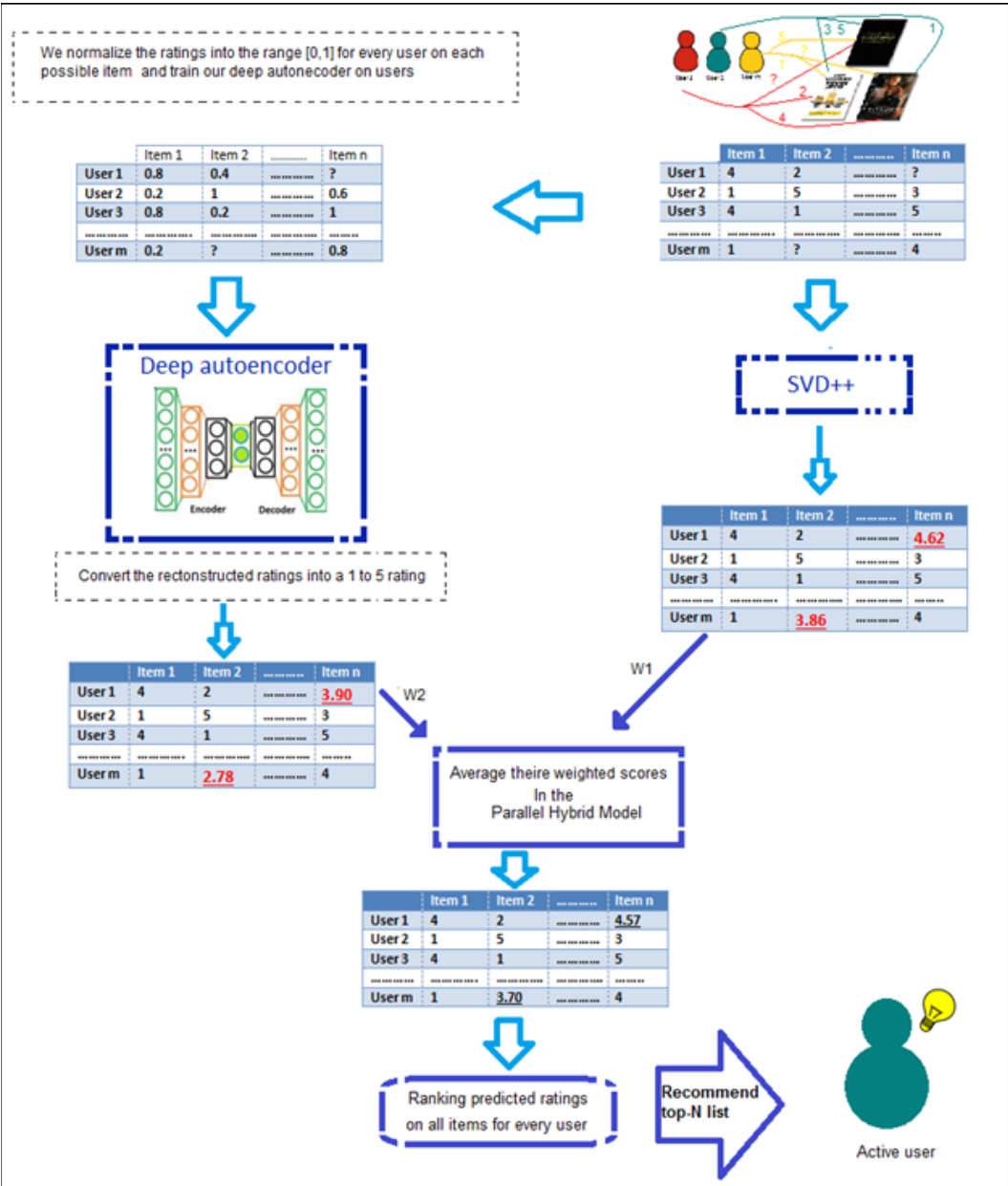
According to the results, we can say that our proposed method has outperformed other methods and we have obtained the lowest RMSE and MAE, regardless the value of the weights. After choosing the optimal weight which is $w = 0.9$, we compare the performance of our proposed method with other hybrid models on the three datasets. Tables 5 and 6 show the final results.

Results in tables 5 and 6 show that our method based on a deep autoencoder and SVD++ outperforms the other hybrid models and gets better results with larger datasets. Thus, the proposed model makes a good solution for high-dimensional data. Note that all methods have large errors on the Book-Crossing dataset as it is extremely rare that people rate a book: less than 5% of the users gave more than 10 different ratings. Therefore, it was not enough data to train a deep network model on.

5.4 Top-n Recommendation List

RMSE and MAE do not matter to a user; what matters is the top-n recommendation list suggested by the recommendation system. Therefore, our model was tested on some randomly selected users from MovieLens 100K dataset that we know their tastes: we choose two random active users (X and Y) who rated more than 30 different movies and analyzed their tastes based on their ratings in the dataset (see table 7). We chose 30 because we wanted to get users who rated one and a half times N , where is N the number of genres ($N = 19$) on MovieLens dataset. One of the random selected users has the id ' $X_{id} = 272$ ' who prefers action, adventure, and science fiction movies. The second user

Figure 3. The framework of our proposed model



with the id ' $Y_{id} = 37$ ' who likes comedy, drama, and romance. See tables 8 and 9 for the final top-10 recommendation list that our hybrid model has built for these two users after removing the movies already watched from the list.

Table 7 shows user X has a rating of at least 4-stars for more than 73% of the action, adventure, and science fiction movies he has already watched. He also watched comedy and children's movies, but only 18% of them had a 4-stars rating. Therefore, we can say that user X likes action, science fiction, and maybe will like a movie that belongs to Action, Sci-Fi, and Comedy simultaneously or perhaps a movie that belongs to Adventure and Children genres. User Y (id = 37) has a rating of at

Figure 4. MAE and RMSE evaluation for the different models. DBN3 is a Deep Belief Network with 3 hidden layers.

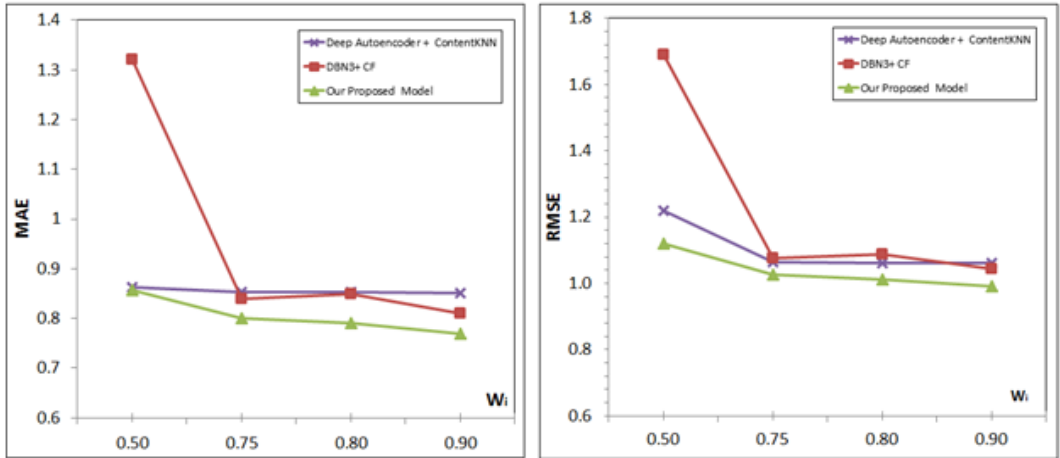


Table 5. RMSE comparison between different models

	Book-Crossing	MovieLens 100K	MovieLens 1M
DAE5 + ContentKNN	3.915	1.060	1.017
DBN3 + CF	3.949	1.043	1.120
Our proposed method	<u>3.726</u>	<u>0.992</u>	<u>0.900</u>

Table 6. MAE comparison between different models

	Book-Crossing	MovieLens 100K	MovieLens 1M
DAE5 + ContentKNN	3.522	0.851	0.813
DBN3 + CF	3.565	0.810	0.913
Our proposed method	<u>3.277</u>	<u>0.777</u>	<u>0.701</u>

Table 7. Description of two randomly selected active users from the MovieLens 100k dataset

	Total Ratings	Most rated genres	Max rating	% of Ratings ≥ 4	Average rating
User X (id=272)	39	Action, Adventure, and Sci-Fi	5	73.33%	4.016
		Comedy and Children	4	18.18%	2.954
User Y (id=37)	32	Romance, Drama and Comedy	5	88.23%	4.412
		Action and Adventure	4	33.33%	3.444

Table 8. Top-10 recommendation list generated by our proposed model for user X (id=272)

Top-10	User X (id=272)		
	Movie name	Genres	Predicted rating
#1	Star Wars: Episode VI - Return of the Jedi (1983)	Action, Adventure and Sci-Fi	4.516
#2	Star Wars: Episode V - The Empire Strikes Back (1980)	Action, Adventure and Sci-Fi	4.485
#3	Godfather: Part II, The (1974)	Crime and Drama	4.475
#4	Back to the Future (1985)	Adventure, Comedy and Sci-Fi	4.470
#5	Inception (2010)	Action, Crime, Drama, Mystery, Sci-Fi, Thriller and IMAX	4.416
#6	Inglourious Basterds (2009)	Action, Drama and War	4.397
#7	Harvey (1950)	Comedy and Fantasy	4.386
#8	It Follows (2014)	Horror	4.378
#9	12 Angry Men (1957)	Drama	4.375
#10	Third Man, The (1949)	Film-Noir, Mystery and Thriller	4.270

Table 9. Top-10 recommendation list generated by our proposed model for user Y (id=37)

Top-10	User Y (id=37)		
	Movie name	Genres	Predicted rating
#1	Father of the Bride Part II (1995)	Comedy	4.857
#2	Before Sunrise (1995)	Drama and Romance	4.794
#3	Jackass Number Two (2006)	Comedy and Documentary	4.530
#4	Happy Endings (2005)	Comedy and Drama	4.499
#5	Mother (1996)	Comedy	4.484
#6	Tex (1982)	Drama	4.399
#7	Snow White and the Seven Dwarfs (1937)	Animation, Children, Drama, Fantasy and Musical	4.221
#8	Next Best Thing, The (2000)	Comedy and Drama	4.154
#9	I.Q. (1994)	Comedy and Romance	4.142
#10	Me and You and Everyone We Know (2005)	Comedy and Drama	4.053

least 4-stars for more than 88% of Romance, Drama and Comedy that he has already watched, but has a rating of 4-stars to only 33% of action and adventure movies he watched. Thus, we can conclude that he likes romance and drama, but he can also watch a movie that combines romance and action, and maybe will be interested in the future to see recommendation of action or adventure movies that he hasn't watched yet.

Based on the results, it is observed that both users will be satisfied by our hybrid recommender system because all the movies in the top-10 recommendation lists are relevant to them. User X likes

action, adventure, and Sci-Fi movies. As expected, our recommender system has suggested the first six movies from the same genres as well as some new ones that might be relevant to him: Horror and Film-Noir which are might be good for two reasons: the first reason is that the model does not take risks because the new suggestions are placed at the end of the list, and the second reason is that a user maybe interested to try new genres due to interest drift. Similarly, based on user Y interests, our model suggested movies that are all in comedy, romance, and drama. In the 7th recommended item on the list, our hybrid model suggested a movie that combines animation and drama which might be desirable for a user that likes drama and sometimes action.

In order to measure the performance in terms of the top-n recommendation list, we compared our hybrid model to two other hybrid models and two methods operating alone (see table 10). The two hybrid models are our deep autoencoder (DAE5) combined with ContentKNN and DBN3 combined with CF. The other two operating alone methods are SVD++ and ContentKNN.

The top-10 recommendation lists suggested by model 1, 3, and 4 are not accurate and mismatch the genres with user interest. For example, user X likes action movies while the models mentioned suggested mainly drama and comedy. Furthermore, we already know that the user has given 4-stars rating for only 18% of drama and comedy that previously watched witch also containing action and adventure. Therefore, these suggestions for this user are irrelevant.

Model 2 (ContentKNN) gave apparently some good results for the six first movies: all are in the same genres list that the user rated before. It is normal since we compute genre-based similarity and release-year based similarity scores in the ContentKNN algorithm for every possible pair of users on the dataset. Next, we look for the nearest neighbor who has the highest similarity score with the selected user. However, there is a problem with this algorithm: it will only recommend movies that the user or his neighbor has watched the same genre, which we can clearly see in the last four movies in the top-10 that all the genres are Western (see table 10). Western is a new genre for the user X which means the given user may or may not like it. However, recommending the same new genre four times in the top-10 will be too risky: we could lose the trust of this user on our recommendation system if he does not like this genre. Therefore, the top-10 list of ContentKNN recommendations is not good.

The proposed model takes advantage of generalization and memorization of feature interactions which are combined in one hybrid model to solve sparsity, high dimensional, and implicit feedback problems. Our deep autoencoder can generalize better to unseen feature combinations through low-dimensional dense embeddings learned from the sparse features while SVD++ can hold the information of correlation between the user-feature and item-feature factors which can be defined as a Memorization of feature interactions. This ensemble approach gave very good results in the top-10 recommendation list: the first two items are action, adventure, and Sci-Fi movies. The model learned that this user loves these genres. Then, we can see a mixture of action, drama, comedy, and Sci-Fi which can be good for a user that watched all these genres. Finally, our hybrid model recommends one new genre at the bottom of the top-10 list which is firstly not risky if the user doesn't like it and secondly, this will encourage him to try out different movies, and he will trust our future recommendation for him. According to the results, we can conclude that our deep autoencoder based hybrid model outperforms all the other mentioned models on the term of RMSE, MAE and top-n recommendation list.

6. CONCLUSION

Deep Learning based recommender systems gain much tractions lately in the machine learning research community. This paper is proposing a hybrid model based on deep autoencoder and SVD++ decomposition method. The hybrid model was tested on the three datasets mentioned above. The result is very promising based on RMSE and MAE. From a user point of view, the analysis of the top-n recommendation list demonstrated that this model can be applied to more practical applications

Table 10.

Top-10	Model1: SVD++		Model2: ContentKNN		Model3: DAE5 + ContentKNN		Model4: DBN3+CF		Our proposed model	
	Movie name	Genre	Movie name	Genre	Movie name	Genre	Movie name	Genre	Movie name	Genre
#1	Godfather, The (1972)	Crime Drama	Black Mask (Hak hap) (1996)	Action Adventure Crime Sci-Fi Thriller	One Magic Christmas (1985)	Drama Fantasy	Shine (1996)	Drama Romance	Star Wars: Episode VI- Return of the Jedi (1983)	Action Adventure Sci-Fi
#2	It happened One Night (1934)	Comedy Romance	Joy Ride (2001)	Adventure Thriller	Step Into Liquid (2002)	Documentary	One Flew Over the Cuckoo's Nest (1975)	Drama	Star Wars: Episode V - The Empire Strikes Back (1980)	Action Adventure Sci-Fi
#3	Graduate, The (1967)	Comedy Drama Romance	What's Up, Tiger Lily? (1966)	Adventure Comedy Crime Thriller	City of God (Cidade de Deus) (2002)	Action Adventure Crime Drama Thriller	James and the Giant Peach (1996)	Adventure Animation Children Fantasy Musical	Godfather: Part II, The (1974)	Crime Drama
#4	Godfather: Part II, The (1974)	Crime Drama	Missing, The (2003)	Adventure Thriller Western	Art of War, The (2000)	Action Thriller	My Fair Lady (1964)	Comedy Drama Musical Romance	Back to the Future (1985)	Action Adventure Comedy Sci-Fi
#5	Tron: Legacy (2010)	Action Adventure Sci-Fi IMAX	City of God (Cidade de Deus) (2002)	Action Adventure Crime Drama Thriller	Taste of Cherry (Ta'm e guilass) (1997)	Drama	Bug's Life, A (1998)	Adventure Animation Children Comedy	Inception (2010)	Action Crime Drama Mystery Sci-Fi Thriller IMAX
#6	African Queen, The (1951)	Adventure Comedy Romance War	24: Redemption (2008)	Action Adventure Crime Drama	King Is Alive, The (2000)	Drama	Princess Bride, The (1987)	Action Adventure Comedy Fantasy	Inglourious Basterds (2009)	Action Drama War
#7	Three Colors: Red (Trois couleurs: Rouge) (1994)	Drama	The Hateful Eight (2015)	Western	Amazing Grace (2006)	Drama Romance	Ben-Hur (1959)	Action Adventure Drama	Harvey (1950)	Comedy Fantasy
#8	Mister Roberts (1955)	Comedy Drama War	Wyatt Earp (1994)	Western	Faust (1926)	Drama Fantasy Horror	Christmas Story, A (1983)	Children Comedy	It Follows (2014)	Horror
#9	Howl's Moving Castle (Hauru no ugoku shiro) (2004)	Adventure Animation Fantasy Romance	True Grit (2010)	Western	Warriors of Virtue (1997)	Action Adventure Children Fantasy	Snow White and the Seven Dwarfs (1937)	Animation Children Drama Fantasy Musical	12 Angry Men (1957)	Drama
#10	Third Man, The (1949)	Film-Noir Mystery Thriller	Shooter, The (1997)	Western	Innocence (2000)	Drama	Wizard of Oz, The (1939)	Adventure Children Fantasy Musical	Third Man, The (1949)	Film-Noir Mystery Thriller

where there are more users than items. Based on the results, the model performs much better with higher dimension datasets.

For future work, the author will expand the model to integrate semantic aspect and context considerations. Therefore, more laboratory testing is needed and more deep learning algorithms will be explored.

FUNDING AGENCY

Publisher has waived the Open Access publishing fee.

REFERENCES

- Bahdanau, D., Cho, K., & Bengio, Y. (2014). *Neural machine translation by jointly learning to align and translate*. arXiv preprint arXiv:1409.0473.
- Beutel, A., Covington, P., Jain, S., Xu, C., Li, J., Gatto, V., & Chi, E. H. (2018). Latent cross: Making use of context in recurrent recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (pp. 46-54). ACM.
- Bougteb, Y., Ouhbi, B., & Frikh, B. (2019). Deep Learning Based Topics Detection. In *2019 Third International Conference on Intelligent Computing in Data Sciences (ICDS)* (pp. 1-7). IEEE.
- Cheng, H., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G.S., Chai, W., Ispir, M., Anil, R., Haque, Z., Hong, L., Jain, V., Liu, X., & Shah, H. (2016). Wide & Deep Learning for Recommender Systems. *DLRS 2016*.
- Côté, M. A., & Larochelle, H. (2016). An infinite restricted Boltzmann machine. *Neural Computation*, 28(7), 1265–1288. doi:10.1162/NECO_a_00848 PMID:27171012
- Fakhfakh, R., Ammar, A. B., & Amar, C. B. (2017). Deep learning-based recommendation: Current issues and challenges. *International Journal of Advanced Computer Science and Applications*, 8, 12.
- Fakhfakh, R., Feki, G., Ammar, A. B., & Amar, C. B. (2016). Personalizing information retrieval: A new model for user preferences elicitation. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (pp. 2091-2096). IEEE.
- Fang, H., Guo, G., Zhang, D., & Shu, Y. (2019). Deep learning-based sequential recommender systems: Concepts, algorithms, and evaluations. In *International Conference on Web Engineering* (pp. 574-577). Springer.
- Guo, H., Tang, R., Ye, Y., Li, Z., & He, X. (2017). *DeepFM: A factorization-machine based neural network for CTR prediction*. arXiv preprint arXiv:1703.04247.
- Hdioud, F., Frikh, B., Ouhbi, B., & Khalil, I. (2017). Multi-Criteria Recommender Systems: A Survey and a Method to Learn New User's Profile. *International Journal of Mobile Computing and Multimedia Communications*, 8(4), 20–48.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778). IEEE.
- Hidasi, B., Karatzoglou, A., Baltrunas, L., & Tikk, D. (2015). *Session-based recommendations with recurrent neural networks*. arXiv preprint arXiv:1511.06939.
- Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4700-4708). IEEE.
- Jing, H., & Smola, A. J. (2017). Neural survival recommender. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (pp. 515-524). ACM.
- Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 426-434). ACM.
- Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8), 30–37.
- Kuchaiev, O., & Ginsburg, B. (2017). *Training deep autoencoders for collaborative filtering*. arXiv preprint arXiv:1708.01715.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436-444.
- Liu, Z., Guo, S., Wang, L., Du, B., & Pang, S. (2019). A multi-objective service composition recommendation method for individualized customer: Hybrid MPA-GSO-DNN model. *Computers & Industrial Engineering*, 128, 122–134.

- Luong, H., Huynh, T., Gauch, S., Do, L., & Hoang, K. (2012). Publication venue recommendation using author network's publication history. In *Asian Conference on Intelligent Information and Database Systems* (pp. 426-435). Springer.
- Ma, W., Liao, X., Dai, W., Pan, W., & Ming, Z. (2021). Holistic Transfer to Rank for Top-n Recommendation. *ACM Transactions on Interactive Intelligent Systems*, 11(1), 1–1. doi:10.1145/3434360
- Moore, J. L., Chen, S., Turnbull, D., & Joachims, T. (2013). Taste Over Time: The Temporal Dynamics of User Preferences. In *ISMIR* (pp. 401-406). Academic Press.
- Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. *ICML*.
- Ouhbi, B., Frikh, B., Zemmouri, E., & Abbad, A. (2018). Deep learning based recommender systems. In *2018 IEEE 5th International Congress on Information Science and Technology (CIST)* (pp. 161-166). IEEE.
- Quadrana, M., Cremonesi, P., & Jannach, D. (2018). Sequence-Aware Recommender Systems. *ACM Computing Surveys*, 51, 1–36.
- Rekik, A., & Jamoussi, S. (2016). Deep learning for hot topic extraction from social streams. In *International Conference on Hybrid Intelligent Systems* (pp. 186-197). Springer.
- Salakhutdinov, R., & Mnih, A. (2008). Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *Proceedings of the 25th international conference on Machine learning* (pp. 880-887). Academic Press.
- Sedhain, S., Menon, A. K., Sanner, S., & Xie, L. (2015). Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th international conference on World Wide Web* (pp. 111-112). Academic Press.
- Song, W., Shi, C., Xiao, Z., Duan, Z., Xu, Y., Zhang, M., & Tang, J. (2019). Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (pp. 1161-1170). ACM.
- Srinivasan, K., Cherukuri, A. K., Vincent, D. R., Garg, A., & Chen, B. Y. (2019). An Efficient Implementation of Artificial Neural Networks with K-fold Cross-validation for Process Optimization. *Journal of Internet Technology*, 20(4), 1213–1225.
- 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* (pp. 11-16). Academic Press.
- Sukhbaatar, S., Szlam, A., Weston, J., & Fergus, R. (2015). End-to-end memory networks. *Advances in Neural Information Processing Systems*, 2440-2448,
- Suriati, S., Dwiastuti, M., & Tulus, T. (2017). Weighted hybrid technique for recommender system. *J. Phys. Conf. Ser.*, 930.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., & Polosukhin, I. (2017). *Attention is All you Need*. ArXiv, abs/1706.03762.
- Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P. A., & Bottou, L. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(12).
- Wang, X., & Wang, Y. (2014). Improving content-based and hybrid music recommendation using deep learning. In *Proceedings of the 22nd ACM international conference on Multimedia* (pp. 627-636). ACM.
- Wei, J., He, J., Chen, K., Zhou, Y., & Tang, Z. (2017). Collaborative filtering and deep learning based recommendation system for cold start items. *Expert Systems with Applications*, 69, 29–39.
- Wu, C. Y., Ahmed, A., Beutel, A., Smola, A. J., & Jing, H. (2017). Recurrent recommender networks. In *Proceedings of the tenth ACM international conference on web search and data mining* (pp. 495-503). ACM.
- Wu, M., Lv, S., Zeng, C., Wang, Z., Zhao, N., Zhu, L., Wang, J., & Wu, M. (2020). A Hybrid Recommender Method Based on Multiple Dimension Attention Analysis. *International Journal of Mobile Computing and Multimedia Communications*, 11(1), 42–57. doi:10.4018/IJCMCMC.2020010103
- Yu, X., Chu, Y., Jiang, F., Guo, Y., & Gong, D. (2018). SVMs classification based two-side cross domain collaborative filtering by inferring intrinsic user and item features. *Knowledge-Based Systems*, 141, 80–91.

Yu, X., Jiang, F., Du, J., & Gong, D. (2019). A cross-domain collaborative filtering algorithm with expanding user and item features via the latent factor space of auxiliary domains. *Pattern Recognition*, 94, 96–109.

Zhou, G., Song, C., Zhu, X., Ma, X., Yan, Y., Dai, X., Zhu, H., Jin, J., Li, H., & Gai, K. (2018). Deep Interest Network for Click-Through Rate Prediction. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Zolaktaf, Z., Babanezhad, R., & Pottinger, R. (2018, April). A generic top-n recommendation framework for trading-off accuracy, novelty, and coverage. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)* (pp. 149-160). IEEE. doi:10.1109/ICDE.2018.00023

Yahya Bougteb is a professor of computer science. He obtained his Master degree in 2018. He is currently a Ph. D. student under the supervision of Prof. Brahim Ouhbi at the National Higher School of Arts and Crafts at MENKES (Morocco). His research is centered on development of deep learning and artificial intelligence algorithms and their application to real word applications such as social media and recommendation systems.

B. Ouhbi received his BS degree in Statistics and Computer science from Mohamed V Science Faculty, Morocco. He obtained his Master degree from Paris VI University in Probability and Applications and PhD degree in Sysrms Control Engineering from Compiègne's Technological University and Habilitation degree in Sysrms Control Science from Mohammadia Engineering School. Currently, His current research interests include data mining, Decision Support system (DSS), Recommender Systems (RS), and web intelligence (IWeb) & Social Networks (SN). He has published over 80 papers in many leading international journals such as *Applied Soft Computing*, *Expert Systems with Applications*, *Communications in Computer and Information Science*, *Studies in computational Intelligence*, *Journal of Intelligent Information Systems*, *Renewable and Sustainable Energy Reviews*, *Inter. Journal of Web Information Systems*, *Journal of mobile and multimedia*, *Stat. Inf. Stoch.*, *Inter. Journal on Artificial Intelligence Tools*, *RAIRO Oper. Res.*, *Statist. and Probab. Lett.*, *Statistical Planning and Inference*, *Inter. J. Perf. Eng.*, *Communications of the ACM*, and others. He was the chair of many sessions in many leading conferences like iiWAS, NGNS, IWAP, CIST. He is a reviewer in many leading international journals. Actually, he is a full Professor at National High School of Arts & Crafts at Moulay Ismail University in Meknès city, Morocco. He has supervised 15 PhD theses on stochastic models for Data mining and Knowledge Discovery. He is a supervisor of 4 PhD theses on Intelligent DSS, RS and Community detection in SN and Deep learning in Big Data context.

Bouchra Frikh is a full professor at the university Sidi Mohamed Benabdellah. She has received her BS from University Mohamed Ben Abdallah in computer science in 1994 and MS from ENSIAS at Rabat in 1999 and PhD in Information Retrieval science in 2003. In 2012, Pr. B. Frikh gets her Habilitation degree from the University Sidi Mohamed Ben Abdallah, Fez, Morocco in Web Mining science. Her research interests include Big Data, Semantic Web, Web mining, Natural language Processing, Recommender systems, Knowledge management, Social networks analysis, Decision support systems in energy and finance sectors. In these areas, she has published over 60 combined refereed journal and conference manuscripts. Pr B. Frikh is a Program Committee member in many international conferences. She is also a reviewer in many international journals. Her scientific papers received many awards in international conferences such as CIST'2014, KEOD 2015 and PAAMS 2016.

Elmoukhtar Zemouri is a professor in Computer Science at ENSAM Meknes, Moulay Ismail University. He holds a PhD (2013, knowledge management in multi-view KDD process) from ENSAM Meknes Morocco, in collaboration with INRIA Sophia-Antipolis France. He received an Engineer degree (2003, computer science and telecommunications) from INPT Rabat, Morocco. His research concerns Machine Learning, in particular Mining and Learning with Graphs and applications. More generally, he is interested in Artificial Intelligence, Deep Learning, knowledge engineering with semantic web technologies.