Journal of Theoretical and Applied Information Technology
31st March 2022. Vol.100. No 6

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

PERFORMANCE COMPARISON OF THREE DIFFERENT TYPES OF AUTOENCODERS USING RECOMMENDATION SYSTEMS

[1] AHED MLEIH AL SBOU, [2] NOOR HAFHIZAH ABD RAHIM

[1]Department of Computer Science, Faculty of Information Technology, Al-Hussein Bin Talal University, Ma'an, Jordan

[2]Faculty of Ocean Engineering Technology and Informatics, Universiti Malaysia Terengganu, Kuala Terengganu, Malaysia

E-mail: [1]ahed_alsbou@ahu.edu.jo, [2]noorhafhizah@umt.edu.my

ABSTRACT

Recommendation system is one of the modern applications to solve information overload problem in a way to provide recommendations of interest to users. Websites such as Amazon, Netflix, Facebook, YouTube, and others apply recommendation system in recommending their products. This also includes recommending news to the readers. However, the systems suffer from some challenges such as high dimensional data, data sparsity, and cold start. To address these problems, deep learning techniques have recently been integrated with recommendation systems and achieve good performance. Autoencoder is one of the most widely used deep learning techniques in recommender systems, especially used for feature extraction, data dimensionality reduction, fast convergence, unsupervised learning, and data reconstruction. In this paper, a performance comparison between three different models of autoencoder is presented which applying in the recommendation systems to further improve the quality of recommendations provided to users. The models are Hybrid Collaborative Recommendation via Semi-AutoEncoder (HRSA), Recommendation via Dual-Autoencoder (ReDa), and Hybrid Collaborative Recommendation method via Dual-Autoencoder (HCRDa). These models work by retrieving the potential latent factors from the sparse rating matrix and predict the missing ratings. The performance is compared based on reconstruction loss that applies to the MovieLens 100K dataset with three different sets of training data: 70%, 80%, and 90%. As a result, it was found that the HCRDa is outperformed the other models in terms of reconstruction loss based on the RMSE evaluation metric and the use of side information in the model. Thus, it is the most effective technique in terms of enhancing the quality of user recommendations.

Keywords: *Autoencoder, Deep learning, Dual-autoencoder, Recommendation systems, Semi-Autoencoder.*

1. INTRODUCTION

In recent years, with the amount of information over the Internet increasing exponentially and rapidly every day, getting relevant information has become a difficult task for the user [1]. Therefore, the recommendation system solves this problem by making suggestions for items that will be useful to the user. Recommendation systems are software tools and methods created to help users find items or services that may interest them and best suit their preferences [2][3].

Recommendation systems can be classified into three methods: content-based filtering (CB), collaborative filtering (CF) [4], and hybrid filtering [5], [6]. CB filtering is based on the features or contents of the items and users' preferences, CF is mainly based on the assumption that similar users have the same tastes and interests, and hybrid filtering is a technique that combines content-based filtering and collaborative filtering to improve performance and overcome the limitations of the previous two approaches [7], [8]. One of the most widely used techniques is collaborative filtering; it is based on analyzing users' past behavior in order to recommend an item to a user based on similarity with other users [9], and it can be grouped into two main categories: memory-based approach (user-based, Item-based) and model-based approach [10]. To make recommendations, the memory-based approaches utilize the whole rating dataset to measure similarities between users or items. On the other hand, Model-based approaches use rating data or some information from a dataset to learn a recommendation model applied by data mining or

machine learning techniques [11], in this paper, a model-based approach is thoroughly investigated.

The most widely used algorithms in the traditional model-based approaches are matrix factorization algorithm, which factorizes the user-item matrix into two low dimensional matrices: the user's feature matrix and the item's feature matrix for personalized recommendation, it is one of the most accurate methods used to reduce the high-level sparsity problem in the recommendation system dataset [9].

In the last decade, deep learning has attracted a lot of attention due to the increased power of computations and processing of large data sets required for training, and its algorithm design is inspired by the function of the human brain [12]. Moreover, it has recently been used in computer vision and natural language processing fields with tremendous success [13], and therefore they have been employed in recommendation systems to overcome some of the challenges that these systems suffer and improve the accuracy and quality of recommendations [14], e.g., Collaborative Filtering Neural network (CFN) [15], deep collaborative filtering (DCF) [16], hybrid collaborative filtering model based on Semi-AutoEncoder (HRSA) [17], Hybrid Collaborative Recommendation method via Dual-Autoencoder (HCRDa) [18] and so on.

In this section, autoencoder networks are examined from deep learning perspective. AutoEncoder is a type of unsupervised learning neural network, which is a feed-forward neural network, consisting of three different types of layers: input layer, hidden layer, and output layer [19]. The first layer is responsible for receiving input data, the middle layers are responsible for performing complex calculations on the input data, and the last layer is responsible for predicting the output. The dimension of the input layer is equal to the dimension of the output layer, although the dimensions of the hidden layer are usually less than them. In which the network receives a vector as input and attempts to reconstruct the input layer at the output layer by using the representation obtained in the hidden layer to match the output to the same vector. The network uses two mappings during the learning process, which are referred to as encoder and decoder, While the encoder maps data from the high-dimensional input layer to the low-dimensional hidden layer, the decoder maps the encoded data from the low-dimensional hidden layer to the output layer to reconstructs the original high-dimensional data [14], [17]. Figure 1 shows a representation of an Autoencoder.

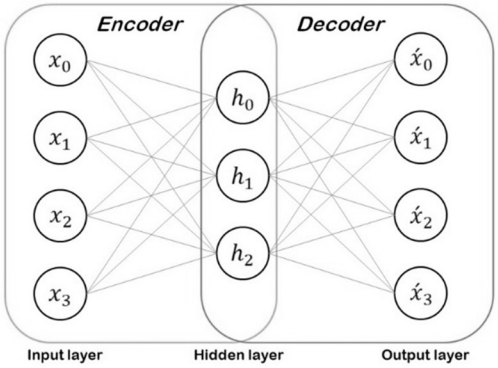

Figure 1: An autoencoder structure.

There are similarities between Autoencoder(AE) and principal component analysis (PCA), both of which are dimension reduction and feature extraction tools [20], [21]. The difference is that the PCA can only map a high dimensional space to a low dimensional coding space in a linear mapping while AE can provide a non-linear mapping from the input layer to the output layer which helps to learn more useful features from PCA [20]. An Autoencoder with only the "code" layer and linear activations should be able to learn PCA transformation in the encoder [22].

Autoencoder has been widely used in recommendations systems, due to its high efficiency in data reconstruction, data dimensionality reduction, and feature extraction. It helps to learn useful features through effective learning of the non-linear relationship present in matrix rating that includes user rating of items and their encoding into data representations [23]. In this paper, three types of Autoencoders will be compared based on the recommendations systems namely :Hybrid Collaborative Recommendation via Semi-AutoEncoder (HRSA), Recommendation via Dual-Autoencoder (ReDa), and Hybrid Collaborative Recommendation method via Dual-Autoencoder (HCRDa) to identify The best method to improve the quality of recommendations to users.

The rest of the article is organized as follows: In Section 2, the architecture of the three types of the Autoencoder is described and Section 3, describes the experiments conducted and also the results obtained in the comparison study. Section 4 a conclusion of the whole study is presented.

Journal of Theoretical and Applied Information Technology
31st March 2022. Vol.100. No 6

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2. ARCHITECTURE OF THE THREE TYPES OF AUTOENCODER BASED ON RECOMMENDATION SYSTEMS

In this section, the architecture of three types of autoencoders used in the recommendation systems is presented.

2.1 Hybrid Collaborative Recommendation Via Semi-autoencoder (HRSA)

2.1.1 Semi-autoencoder structure

In [17], a new framework was proposed, called Semi-AutoEncoder, which is based on AutoEncoder, which is employed in the collaborative filtering model to predict ratings and make top-n recommendations. It uses content information to aid in the flexible learning of feature representations or reconstructions.

AutoEncoder is a feed-forward and unsupervised learning neural network. Generally, it contains three layers: input, hidden, and output. The dimensions of the input and output layer are the same, but the dimensions of the hidden layer are less than them. However, it is not necessary that the dimensions of the input and output layer be the same, as in the first case, the size of the input layer dimensions is longer than the size of the output layer dimensions as described in Figure 2 or the second case, and the size of the output layer dimensions is longer than the size of the input layer dimensions as described in Figure 3. In the first case, it makes it simple to include more information in the input layer, and by sampling various subsets of the inputs; this approach can easily capture different representations and reconstruction. In the second case, allows it to create new items from the middle layer. The first case is referred to as a Semi-AutoEncoder by [17].

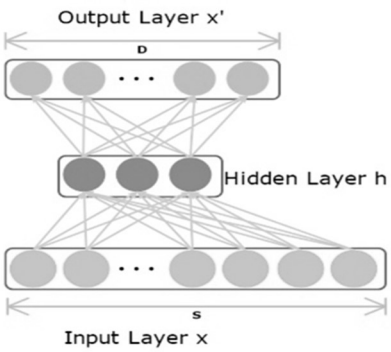

Figure 2: case1.
Source: Adapted from [17]

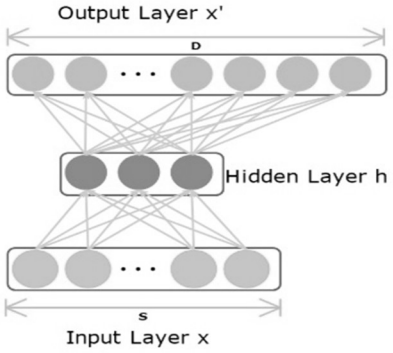

Figure 3: case2.
Source: Adapted from [17]

The Semi-AutoEncoder mainly consists of three layers: the input layer, the hidden layer, and the output layer, each with dimensions as shown in Figure2:$x \in R^S, h \in R^H, \acute{x} \in R^D$, respectively, where H < D < S. During the training of the Semi-AutoEncoder, the output $\acute{x}$ must be matched to a subset of the input to calculate the loss function, thus extracting a subset of the input x with the same output $\acute{x}$ dimensions and denotes it as sub (x). Network encoding and decoding map and loss function as shown in Eq. (1), Eq. (2), and Eq. (3):

$$h = g(Wx + b) \quad (1)$$

Where g is a nonlinear activation function, $W \in R^{H \times S}$ is a weight matrix,$b \in R^H$ is a bias. After that, a decoding mapping is used to map the latent representation h back to a reconstruction $\acute{x}$:

$$\acute{x} = f(\acute{W}h + \acute{b}) \quad (2)$$

Where f is a nonlinear activation function, $\acute{W} \in R^{D \times H}$ is a weight matrix,$\acute{b} \in R^D$ is a bias. The reconstruction error is minimized to optimize the parameters of a Semi-AutoEncoder, as follows:

$$\mathcal{L}(x, \acute{x}) = \|sub(x) - \acute{x}\|^2 \quad (3)$$

2.1.2 Semi-autoencoder for collaborative filtering

In [11], the Semi-AutoEncoder has been used to improve the performance of recommendations systems in the two tasks of rating predictions and providing top-n recommendations. By using the benefit of Semi-AutoEncoder capabilities to easily integrate user-profiles and item information into collaborative filtering. The method in the first task, which is rating the predictions, is compared with other methods of the same type.

Figure 4 shows the structure of the Semi-AutoEncoder model used to predict ratings. Where $c^i \in R^K$ $(i = 1, \dots, N)$ to denote the feature of item i. For each item i, there is an observed partial r^i vector and a c^i feature vector. These two vectors are combined and referred to as a $cat(r^i; c^i) \in R^{M+K}$ $(i = 1, \dots, N)$. All features of N items are represented by a capital letter using $C^I \in R^{N\times K}$ and the concatenate vectors are represented using $cat(r^I; C^I) \in R^{N\times(M+K)}$. Next, the concatenated vector is used as the input, giving the hidden representation h through an encoding mapping as shown in Eq. (4).

$$h(r^I; C^I) = g(cat(r^I; C^I).W + b) \quad (4)$$

Where $W \in R^{(M+K)\times H}$ is the weight matrix, $b \in R^H$ is a bias vector. After that, a decoding mapping is used to map the latent representation h back to a reconstruction $\acute{r}$ as shown in Eq. (5):

$$\acute{r}^I = f(h(r^I; C^I).\acute{W} + \acute{b} \quad (5)$$

Where $W \in \acute{R}^{H\times M}$ is the weight matrix, $\acute{b} \in R^M$ is a bias vector. The goal is to reduce the reconstruction error by comparing the output $\acute{r}^I$ with the subset of the input, where $(x) = r^i$, through the objective function as follows:

$$\underset{W,W,b,\acute{b},\acute{r}^{ui}\in\Omega}{arg\,min} \frac{1}{N}\sum_{i=1}^{N}\left\|r^i - \acute{r}^i\right\|_2^2 + \frac{\gamma}{2}(\|w\|_2^2 + \|\acute{w}\|_2^2) \quad (6)$$

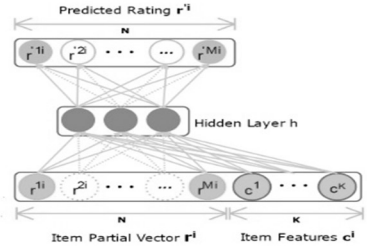

Figure 4: The graphic model of HRSA: Rating prediction. Source: Adapted from [17]

2.2 Recommendation Via Dual-Autoencoder (ReDa)

In [24] model it is mainly based on a technique used in traditional recommendation systems, which are matrix factorization technique, and which decomposes the rating matrix into two low-dimensional matrices, a user-specific matrix, and an item-specific matrix, and then makes further predictions using the factorized matrices. Due to the tremendous success provided by deep learning in multiple domains such as image and video processing, deep learning methods have been used to get a better representation of the latent factors of users and items for recommendations.

In [24] technique depends on pair of Autoencoders synchronously called the Dual-Autoencoder, one Autoencoder learns new hidden representations of the user and the other learns new hidden representations of the item, The learned representations of users and items are used to reduce the differences in training data. In Figure 5 the structure of Dual-Autoencoder is shown. Where $R \in \mathbb{R}^{m\times n}$ to denote rating matrix or check-in matrix, where m and n are the numbers of users and items, respectively, $R_1 = R, R_2 = R^T$, T denotes the transposition of a matrix $\xi_1 \in \mathbb{R}^{k\times n}$ is the item's latent representation, $\xi_2 \in \mathbb{R}^{k\times m}$ is the user's latent representation, $W_1 \in \mathbb{R}^{k\times m}, b_1 \in \mathbb{R}^{n\times 1}$ are weight matrix and bias vector of encoding for learning the latent representations of items, respectively, $\acute{W}_1 \in \mathbb{R}^{m\times k}, \acute{b}_1 \in \mathbb{R}^{n\times 1}$ are weight matrix and bias vector of decoding for learning the latent representations of items, respectively. $W_2 \in \mathbb{R}^{k\times n}, b_2 \in \mathbb{R}^{m\times 1}$ are weight matrix and bias vector of encoding for learning the latent representations of users, respectively, $\acute{W}_2 \in \mathbb{R}^{n\times k}, \acute{b}_2 \in \mathbb{R}^{m\times 1}$ are weight matrix and bias vector of decoding for learning the latent representations of users, respectively.

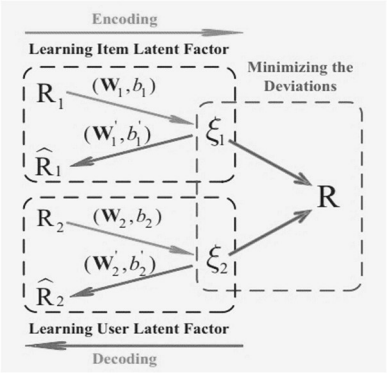

Figure 5: ReDa structure (R is a rating matrix, $R_1 = R, R_2 = R^T$). Source: Adapted from [24]

The main aim behind matrix factorization-based recommendation algorithms is to obtain excellent latent factors between items and users. In [24], the Autoencoders technique is used to learn the latent factors of user and item synchronously and to obtain better latent representations. To learn the latent factors of the user by the Autoencoder, each

user in the rating matrix can be considered as an input instance, and the items can be considered as the features which correspond to it. On the other hand, to learn the latent factors of the item by the Autoencoder, each item in the rating matrix can be considered as an input instance, and the users can be considered as the features which correspond to it.

The encoding and decoding operations to learn latent representations of items are as follows:

$$\xi_1 = f(W_1 R_1 + e_1 b_1^T) \quad (7)$$

$$\hat{R}_1 = f(\acute{W}_1 \xi_1 + \acute{e}_1 \acute{b}_1^T) \quad (8)$$

Where $e_1 \in R^{k\times 1}$ and $\acute{e}_1 \in R^{m\times 1}$ are two constant vectors, each with one entry, f denotes the nonlinear activation function (the sigmoid function).

The following is the objective function for learning the latent representations of items:

$$\mathcal{J}_b = \left\| I_1 o(R_1 - \hat{R}_1 \right\|^2 \quad (9)$$

Where $I_{1=}I$, $I \in R^{m\times n}$ is the indicator matrix, if the user rates the item $R_{ij} \neq 0, I_{ij} = 1$ otherwise then $I_{ij} = 0$, $\circ$ is the element-wise product of vectors or matrices. Similarly, the encoding and decoding operations to learn latent representations of users are as follows:

$$\xi_2 = f(W_2 R_2 + e_2 b_2^T) \quad (10)$$

$$\hat{R}_2 = f(\acute{W}_2 \xi_2 + \acute{e}_2 \acute{b}_2^T) \quad (11)$$

Where $e_2 \in R^{k\times 1}$ and $\acute{e}_2 \in R^{n\times 1}$ are two constant vectors, each with one entry, f denotes the non-linear activation function (the sigmoid function).

The following is the objective function for learning the latent representations of users:

$$\mathcal{J}_a = \left\| I_2 o(R_2 - \hat{R}_2 \right\|^2 \quad (12)$$

Where $I_{2=}I^T$ is the transposition matrices of I.

To reduce deviations of training data, latent representations of users and items learned from Autoencoders are combined into a single matrix and subtracted from the rating matrix as in the following formula:

$$\mathcal{J}_c = \left\| Io(R - \xi_2^T \xi_1 \right\|^2 \quad (13)$$

Lastly, the framework's whole optimization problem is:

$$\mathcal{J} = \mathcal{J}_c + \alpha.\mathcal{J}_b + \beta.\mathcal{J}_a + \gamma.\left(\|W_1\|^2 + \|b_1\|^2 + \left\|\acute{W}_1\right\|^2 + \left\|\acute{b}_1\right\|^2\right) + \|W_1\|^2 + \|b_1\|^2 + \left\|\acute{W}_1\right\|^2 + \left\|\acute{b}_1\right\|^2 \quad (14)$$

Where α, β, γ are trade-off parameters, and the fourth point is framework parameter regularization. To obtain the optimal solution, gradient descent methods were used. To see the mathematical derivation method for the equation optimization problem, see [24].

2.3 Hybrid Collaborative Recommendation method via Dual-Autoencoder (HCRDa)

In the above two approaches, they suffer from some problems that may affect their performance in the recommendation systems, namely: the output of the Autoencoder was used to directly predict the missing values in the recommender systems. Furthermore, an Autoencoder's parameters must be pre-trained ahead of time, significantly increasing the time complexity [18]. In HCRDa, more problems will be addressed.

In [18], the approaches of Dual-Autoencoder, Semi-Autoencoder, and Matrix Factorization are combined into a novel technique called Hybrid Collaborative Recommendation method via Dual-Autoencoder (HCRDa). The Dual-Autoencoder was used to simultaneously learn user and item feature representations, with the Semi-Autoencoder, additional information for users and items is combined to make hybrid recommendations, and by integrating matrix factorization into the Autoencoder's training process, the quality of hidden features for items and users is enhanced even more. As shown in Figure 6.

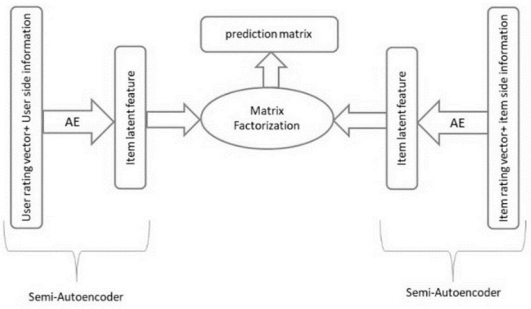

Figure 6: integrating matrix factorization into the Autoencoder's training process.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

In this section, the methodology of this approach is further discussed. Figure 7 illustrates the structure of [18] proposed approach. Where $R \in R^{n\times m}$ is the rating matrix, n is the number of items and m is the number of users, r^{ui} denotes that user $u \in \{1 \dots m\}$ gave the item $i \in \{1 \dots n\}$ a rating, the rating matrix's row is denoted by r^i , the rating matrix's column is denoted by r^u , user u's attribute features and item i's attribute features are represented by a^u, a^i, respectively. The latent representation of users or items is referred to as ξ.

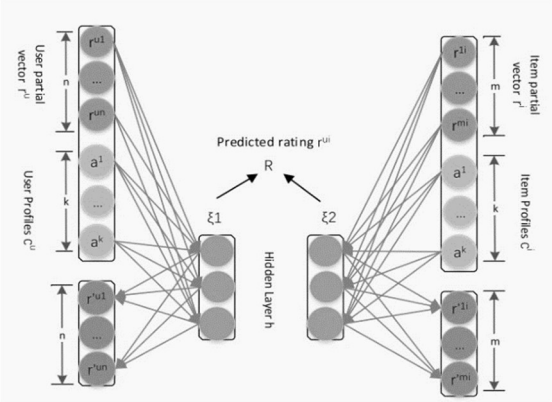

Figure 7: structure of HCRDa.
Source: Adapted from [18]

The input of the Autoencoder to learn the representation of the hidden feature of the item i is the additional information vector of the item a^i and the rating vector r^i that are combined. The concatenation of r^i and a^i are denoted by $In(r^i, a^i) \in R^{(m+y_i)}$, where y_i denotes the item's number of attributes, and the concatenation of r^I and A^I are denoted by $In(r^I, A^I) \in R^{n\times(m+y_i)}$, where r^I denotes the partial observed vectors for all of the items, and A^I denotes the additional features of all items. To learn the hidden representations of the item ξ_i, the $In(r^I, A^I)$ is input into the Semi-Autoencoder. Eq. (15) and Eq. (16) represent the item's encoding and decoding mappings respectively.

$$\xi_i = f(In(r^I, A^I)W + b) \quad (15)$$

$$\acute{r}^I = g(\acute{W}\xi_i + \acute{b}) \quad (16)$$

Where ξ_i is an encoding map to learn the latent representations of the item i, $W \in R^{(m+y_i)\times h}$ and $\acute{W} \in R^{h\times m}$ are the weight matrices, $b \in R^h$ and $\acute{b} \in R^m$ are bias vectors, the hidden layer's dimensions are denoted by h, and nonlinear functions f is sigmoid function and nonlinear functions g is the identity function. The purpose of the Semi-Autoencoder is to approximate the partial input, which is the rating data, into the output. As a result of a Semi-Autoencoder that handles the item reconstruction loss, the objective function is f_i in Eq. (17).

$$f_i = \min_{W,\acute{W},b,\acute{b},r^{ui}\in\Omega} = \|r^I - \acute{r}^I\|^2 \quad (17)$$

Where Ω is the set of observed ratings.

Similarly, Autoencoder input for learning the hidden feature representation of user u is the additional information vector of the user a^u and the rating vector r^u that are combined. The concatenation of r^u and a^u are denoted by $In(r^u, a^u) \in R^{(n+y_u)}$, where y_u denotes the user's number of attributes, and the concatenation of r^U and A^U are denoted by $In(r^U, A^U) \in R^{m\times(n+y_u)}$, where r^U denotes the partial observed vectors for all of the users, and A^U denotes the additional features of all users. To learn the hidden representations of the user ξ_u, the $In(r^U, A^U)$ is input into the Semi-Autoencoder. Eq. (18) and Eq. (19) represent the user's encoding and decoding mappings respectively.

$$\xi_u = f(In(r^U, A^U)W_u + b_u) \quad (18)$$

$$\acute{r}^U = g(\acute{W}_u\xi_U + \acute{b_u}) \quad (19)$$

Where ξ_u is an encoding map to learn the user's latent representations u, $W_u \in R^{(n+y_u)\times h}$ and $\acute{W}_u \in R^{h\times n}$ are the weight matrices, $b_u \in R^h$ and $\acute{b}_u \in R^n$ are bias vectors, the hidden layer's dimensions are denoted by h, and nonlinear functions f is sigmoid function and nonlinear functions g is the identity function. The purpose of the Semi-Autoencoder is to approximate the partial input, which is the rating data, into the output. As a result of a Semi-Autoencoder that handles the user reconstruction loss, the objective function is f_u in Eq. (20).

$$f_u = \min_{W_u,\acute{W}_u,b_u,\acute{b},_u r^{ui}\in\Omega} = \|r^U - \acute{r}^I\|^2 \quad (20)$$

The approach of matrix factorization is combined with the autoencoder learning process in order to better learn the latent factors of items and users at the same time, in order for the final prediction matrix $\xi_i{\xi_u}^T$ to be close to the actual rating matrix, where T is a transposition of the matrix, and the third term of the objective function is Eq. (21).

$$f_{mf} = \min_{W,\acute{W},b,\acute{b},W_u,\acute{W}_u,b_u,\acute{b}_u r^{ui}\in\Omega} = \left\|R - \xi_i {\xi_u}^T\right\|^2 \quad (21)$$

The regularization component from the weight matrix of the $l_2\ norm$ is added as a term to the last objective function to prevent the problem of overfitting, as stated in Eq. (22).

$$f_l = \frac{\gamma}{2}\left(\|W\|^2 + \left\|\acute{W}\right\|^2 + \|W_u\|^2 + \left\|\acute{W}_u\right\|^2\right) \quad (22)$$

Eq. (23) shows the model framework's final objective function:

$$\min_{W,\acute{W},b,\acute{b},W_u,\acute{W}_u,b_u,\acute{b}_u r^{ui}\in\Omega} f_i + f_u + f_{mf} + f_l \quad (23)$$

The proposed method for optimizing Eq. (23) employs stochastic gradient descent. Until the algorithm reaches a point of convergence.

.

3. EXPERIMENTS AND RESULTS

In this section, a group of experiments are conducted to compare the three different techniques used in Autoencoder. In the HRSA model [11], an experiment was conducted on Movielens 100K and Movielens 1M datasets, in the ReDa model [17], the experiment was conducted on MovieLens 100K, MovieLens 1M, Douban Movie, and Douban Book datasets, and in the HCRDa model [12] the experiment was conducted on Movielens 100K, MovieTweetings, FilmTrust datasets.

The Movielens 100K dataset is utilized in all three models; therefore, this database is adopted as a dataset to compare the three models, which one is better. It has 943 users and 1682 movies with a total of 100,000 ratings and a rating density of 6.3%. The explicit rating scale in this dataset are evaluated using a range of (1 to 5) stars.

The quality of the Autoencoder model can be evaluated utilizing different kinds of metrics that can be precise or coverage, such as root mean square error (RMSE) [15], [25], mean error (MAE) [26], recall [16]. RMSE is the most common and most widely used, it is a measure of the deviation of predicted values from the user's observed values and the smaller the RMSE, the better the performance of the model [27]. As shown in Eq. (24), the RMSE of the model is formally calculated, where n denotes the number of observations, $\hat{y}_i$ denotes the predicted values, y_i denotes the observed values. There are several metrics used to evaluate the prediction performance of each model, including RMSE, MAE, and recall. The RMSE is the measure available in the three models. Therefore, the RMSE is adopted here for the purpose of comparison.

$$RMSE = \sqrt{\frac{1}{n}\sum_1^n (\hat{y}_i - y_i)^2} \quad (24)$$

The three models were evaluated by taking a random sample with different percentages of rating records as a training set, and the rest was used as a test set. For the aim of comparing the three models, a standardized percentage will be adopted, which is 70%, 80%, and 90% of the data set sample as a training set and the remaining 30%, 20%, and 10% as a test set, respectively. Training a model requires a set of hyper-parameters by which it can achieve the best performance, and each model has its own hyper-parameters values. In an HRSA model, the learning rate was set to 0.001 and the regularization rate γ to 1 and 500 is the size of the hidden neural network. In the ReDa model, it is set to 0.5, 0.5, and 1 as the trade-off parameters β, α, and γ respectively, and 50 is the size of the hidden neural network. In the HCRDa model, the learning rate and the number of hidden neurons is the same as in HRSA, but the regularization rate γ = 1. It is noticed that there are similar and different parameters between the three models in the training, among which were mentioned: HRSA and HCRDa each have the same activation function used to map the encoding and the decoding, which is sigmoid and identity respectively. But in ReDa the activation function in the encoder mapping is the same as in decoder mapping and is sigmoid. Also, HRSA and HCRDa have the same number of hidden layer neurons while ReDa is different from them .HRSA and HCRDa include side information for users and items to improve performance and avoid a cold start problem, whereas ReDa does not. Both HCRDa and ReDa are optimized using the stochastic gradient descent method, whereas HRSA is optimized using the Adam method.

Table 1 shows the RMSE scale for the three models, which was obtained from the reported papers for both HCRDa and ReDa due to both did not contain the source code. Whereas, the HRSA model has the source code[1] and we implemented it by Google Colab[2] in order to extract the results. Based on the results, it can be concluded that HCRDa on 80% and 90% of training data outperforms both HRSA and ReDa in terms of

1 https://github.com/cheungdaven/semi-ae-recsys

2 https://colab.research.google.com/

reconstruction loss, whereas HRSA with 70% of training data outperforms both HCRDa and ReDa, its accuracy measure improves slightly with increasing training data size. The performance of the three models in the Movielens-100k data set shown in the graph in Figure 8 reveals that HCRDa is effective as indicated by the decreasing RMSE value with increasing training data. The HCRDa model on 90% of the training data improved the performance of the RMSE assessment criteria compared to the HRSA and ReDa models by 11.5% and 13.6%, respectively. It is also noted that both the HCRDa and HRSA models include side information in their experiments, whereas the ReDa model does not include side information in their experiment, therefore the performance of HCRDa and HRSA is better than ReDa due to the effect of the side information on the results. The parameters of the Autoencoder must be pre-trained in both HRSA and ReDa, which considerably increases the time complexity. The Autoencoder in HCRDa learns feature representations of items and users at the same time, reducing complexity time. Through this comparison, it can be determined that the HCRDa model outperforms the other two models in performance.

Table 1: Average RMSE on Movielens 100k with 70%, 80%, and 90% of training data.

Model	Training data of Movielens-100K		
	70%	80%	90%
HRSA	0.906± 0.004	0.896 ± 0.003	0.890± 0.007
ReDa	0.9231± 0.0081	0.9190± 0.0056	0.9114± 0.0093
HCRDa	0.9176±0.0 009	0.8645± 0.0010	0.7879± 0.0061

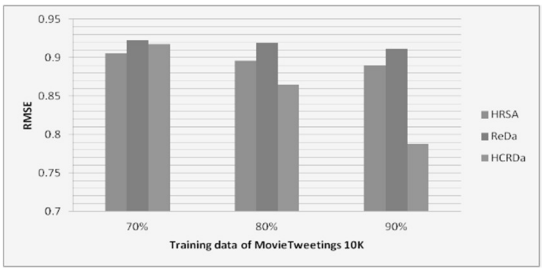

Figure 8: The performance of the three models on the Movielens-100k dataset.

4. CONCLUSION

In this paper, the aim was to assess the performances of three types of Autoencoder models that apply to the recommendation systems. The models are HRSA, ReDa, and HCRDa. Each of the models have been explained in details. In this investigation, MovieLens 100K dataset has been applied to the models. The results of the experiments show that HCRDa has the lowest reconstruction error (RMSE), which outperformed others. Moreover, the models with side information are also performed better than others, for example HCRDa and HRSA. For future works, to improve the performance in term of accuracy, more experiments will be carried out such as adding more network layers to the model and include more explicit or implicit side information to the model. This would be a fruitful area for further work in recommendation systems.

ACKNOWLEDGEMENT

This research was supported by the fundamental research grant scheme for research acculturation of early career researchers (FRGS-RACER) with the reference code of RACER/1/2019/ICT02/UMT//2 and vote number 59547 under the Malaysia Ministry Of Education (RACER 2019-1).

REFRENCES:

[1] A. Singhal, P. Sinha, and R. Pant, "Use of Deep Learning in Modern Recommendation System: A Summary of Recent Works," *Int. J. Comput. Appl.*, vol. 180, no. 7, pp. 17–22, 2017, doi: 10.5120/ijca2017916055.

[2] S. Jain, A. Grover, P. S. Thakur, and S. K. Choudhary, "Trends, problems and solutions of recommender system," *Int. Conf. Comput. Commun. Autom. ICCCA 2015*, pp. 955–958, 2015, doi: 10.1109/CCAA.2015.7148534.

[3] A. SAMIH, A. GHADI, and A. FENNAN, "Deep graph embeddings in recommender systems: A survey," *J. Theor. Appl. Inf. Technol.*, vol. 99, no. 15, pp. 3812–3823, 2021, doi: 10.5281/zenodo.5353504.

[4] S. Sharma, V. Rana, and M. Malhotra, "Automatic recommendation system based on hybrid filtering algorithm," *Educ. Inf. Technol.*, no. 0123456789, 2021, doi: 10.1007/s10639-021-10643-8.

[5] A. Dhruv, A. Kamath, A. Powar, and K. Gaikwad, "Artist Recommendation System Using Hybrid Method: A Novel Approach," in *Emerging Research in Computing, Information, Communication and Applications*, Springer Singapore, 2019, pp. 527–542.

[6] R. Hooda, K. Singh, and S. Dhawan, "A Study of Recommender Systems on Social Networks

and Content-based Web Systems," *Int. J. Comput. Appl.*, vol. 97, no. 4, pp. 23–28, 2014, doi: 10.5120/16996-7128.

[7] Li Qing and Byeong Man Kim, "Clustering approach for hybrid recommender system," *Proc. IEEE/WIC Int. Conf. Web Intell. (WI 2003)*, pp. 33–38, 2003, doi: 10.1109/WI.2003.1241167.

[8] M. Nilashi, O. Ibrahim, and K. Bagherifard, "A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques," *Expert Syst. Appl.*, vol. 92, pp. 507–520, 2018, doi: 10.1016/j.eswa.2017.09.058.

[9] D. Bokde, S. Girase, and D. Mukhopadhyay, "Matrix Factorization model in Collaborative Filtering algorithms: A survey," *Procedia Comput. Sci.*, vol. 49, no. 1, pp. 136–146, 2015, doi: 10.1016/j.procs.2015.04.237.

[10] Y. Dou, H. Yang, and X. Deng, "A Survey of Collaborative Filtering Algorithms for Social Recommender Systems," *Proc. - 2016 12th Int. Conf. Semant. Knowl. Grids, SKG 2016*, pp. 40–46, 2017, doi: 10.1109/SKG.2016.014.

[11] F. Anwar, N. Iltaf, H. Afzal, and H. Abbas, "A Deep Learning Framework to Predict Rating for Cold Start Item Using Item Metadata," pp. 313–319, 2019, doi: 10.1109/wetice.2019.00071.

[12] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep Learning Based Recommender System," *ACM Comput. Surv.*, vol. 52, no. 1, pp. 1–38, Feb. 2019, doi: 10.1145/3285029.

[13] A. Bashar, "SURVEY ON EVOLVING DEEP LEARNING NEURAL NETWORK," no. July, 2020, doi: 10.36548/jaicn.2019.2.003.

[14] Z. Batmaz, A. Yurekli, A. Bilge, and C. Kaleli, "A review on deep learning for recommender systems: challenges and remedies," *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 1–37, Jun. 2019, doi: 10.1007/s10462-018-9654-y.

[15] F. Strub, J. Mary, and R. Gaudel, "Hybrid Collaborative Filtering with Autoencoders," 2016, [Online]. Available: http://arxiv.org/abs/1603.00806.

[16] S. Li, J. Kawale, and Y. Fu, "Deep collaborative filtering via marginalized denoising auto-encoder," *Int. Conf. Inf. Knowl. Manag. Proc.*, vol. 19-23-Oct-, pp. 811–820, 2015, doi: 10.1145/2806416.2806527.

[17] S. Zhang, L. Yao, X. Xu, S. Wang, and L. Zhu, "Hybrid collaborative recommendation via Semi-AutoEncoder," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10634 LNCS, pp. 185–193, 2017, doi: 10.1007/978-3-319-70087-8_20.

[18] B. Dong, Y. Zhu, L. Li, and X. Wu, "Hybrid Collaborative Recommendation via Dual-Autoencoder," *IEEE Access*, vol. 8, pp. 46030–46040, 2020, doi: 10.1109/ACCESS.2020.2979255.

[19] V. Kumar, "Deep Learning as a Frontier of Machine Learning : A Review Deep Learning as a Frontier of Machine Learning : A Review," no. July 2018, 2019, doi: 10.5120/ijca2018917433.

[20] C. C. Tan and C. Eswaran, "Performance comparison of three types of autoencoder neural networks," *Proc. - 2nd Asia Int. Conf. Model. Simulation, AMS 2008*, pp. 213–218, 2008, doi: 10.1109/AMS.2008.105.

[21] S. Guide, I. Deep, L. Models, and T. B. Ii, *Introduction to Deep Learning Using R Introduction to Deep*. .

[22] O. Kuchaiev and B. Ginsburg, "Training Deep AutoEncoders for Collaborative Filtering," 2017, [Online]. Available: http://arxiv.org/abs/1708.01715.

[23] G. Zhang, Y. Liu, and X. Jin, "A survey of autoencoder-based recommender systems," *Front. Comput. Sci.*, vol. 14, no. 2, pp. 430–450, 2020, doi: 10.1007/s11704-018-8052-6.

[24] F. Zhuang, Z. Zhang, M. Qian, C. Shi, X. Xie, and Q. He, "Representation learning via Dual-Autoencoder for recommendation," *Neural Networks*, vol. 90, pp. 83–89, 2017, doi: 10.1016/j.neunet.2017.03.009.

[25] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec," pp. 111–112, 2015, doi: 10.1145/2740908.2742726.

[26] Y. Ouyang *et al.*, "Autoencoder-based collaborative filtering," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8836, pp. 284–291, 2014, doi: 10.1007/978-3-319-12643-2_35.

[27] F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egypt. Informatics J.*, vol. 16, no. 3, pp. 261–273, 2015, doi: 10.1016/j.eij.2015.06.005.