# Tavsiye Sistemlerinde Derin Otokodlayicilar: Nesnelerin Interneti Hizmet Tavsiyesi Üzerine Bir Uygulama

*Araştırma Makalesi/Research Article*

Şule BİRİM

Salihli İktisadi ve İdari Birimler Fakültesi, Manisa Celal Bayar Üniversitesi, Manisa, Türkiye
sule.ozturk@cbu.edu.tr

(Geliş/Received:07.02.2020; Kabul/Accepted:01.07.2021)
DOI: 10.17671/gazibtd.685500

***Özet***— Nesnelerin İnterneti (IoT) küçük ve birbirine bağlı, internet aracılığıyla bilgi paylaşımı sağlayan cihazlardan oluşan bir sistemdir. Gelecek IoT cihazlarının gerçekleştirdiği hizmetlerin sayısının oldukça artmasını beklemektedir. Bu nedenle IoT hizmeti önerme, gelecekte çok önemli bir faaliyet halini alacaktır. Bu çalışma, IoT hizmetleri tavsiyesi için en iyi yöntemi bulmayı hedeflemektedir. Bu amaç doğrultusunda bu çalışma derin otokodlayıcılar yöntemi ile, kullanıcılara IoT hizmet ve uygulamalarını kullandıkları cihazlara bağlı olarak önermeyi amaçlamıştır. Derin otokodlayıcılar, kullanıcı hizmet tercih matrisini tahmin edebilmek için sinir ağlarını kullanır. Bu çalışmada kullanılan veri, gerçek bir Sosyal IoT veri setinden yararlanılarak oluşturulmuştur. Sonuçlara göre derin otokodlayıcılar, teknoloji harikası öneri yöntemlere göre daha başarılı bir performans ortaya koymuştur. Bulunan sonuçlara göre Derin otokodlayıcılar diğer yöntemlere göre performans göstergelerinde %13.5 ile %69.5 arasında değişen oranlarda iyileştirme sağlamıştır. Bu sonuca ek olarak ELU (exponential linear units), özel bir çeşit aktivasyon fonksiyonu ile oto kodlayıcıların performansının arttırılabileceği görülmüştür. Bu çalışma IoT hizmet önerisi literatürüne geleneksel ve teknoloji harikası kabul edilen yöntemlere bir alternatif sunarak katkı sağlamaktadır.

***Anahtar Kelimeler***— nesnelerin interneti (iot), öneri sistemleri, hizmet önerisi, işbirlikçi filtreleme, derin otokodlayıcılar, sinir ağları

# Deep AutoEncoders in Recommender Systems: An Application about Internet of Things Service Recommendation

***Abstract***— The Internet of Things (IoT) is a system that includes small interconnected devices sharing information through the Internet. Future expects an increasing number of services in IoT devices. Therefore, recommending IoT services will be a vital task for the future of IoT and the convenience of the users. This study aims to find the best methodology to provide IoT service recommendation. With this aim, this study proposes deep autoencoders methodology to recommend services and applications to users based on the devices they own. Deep autoencoders utilize neural networks to predict user service preference matrix. The data used in this study is constructed from a real-world Social IoT dataset. The results showed that deep autoencoders outperformed the state-of-the-art recommendation methods. According to the results, Deep autoencoders improved performance indicators varying between 13.5% and 69.5% compared to other methods. Findings also indicate that the performance of the deep autoencoders can be enhanced by using ELU (exponential linear units), a specific type of activation function. This study contributes to the IoT service recommendation literature by proposing a superior approach when compared to the traditional recommendation techniques.

***Keywords***— the internet of things (iot), recommendation systems, service recommendation, collaborative filtering, deep autoencoders, neural networks

## 1. INTRODUCTION

In recent times technological innovations allow people to communicate with small devices over the internet. The small interconnected devices that share information based on the standard communication protocols of the Internet generate the system of The Internet of Things (IoT). The IoT devices people use in everyday life include smartphones, power meters, smart watches, temperature meters and any sensor including devices that stores and shares data. Future expects that a massive amount of data will be exchanged over the Internet between the IoT using parties [1]. IoT is expected to be widespread in such a way that in the future amount of data transmitted between the devices will be much higher than the amount of data transferred between humans and machines [2]. A study claims that the number of devices connected to the Internet will double between the years 2016 – 2020 and will increase from 22.9 billion to 50 billion [3].

As the IoT becoming more popular the amount and the diversity of services and applications used in IoT devices increases dramatically. In cities, to make the traffic smarter and more secure, units of the traffic interact with each other better with the help of IoT [4]. IoT is also widely used in digital payments in recent years [5]. Many companies are offering a vast amount of services to IoT users. Telus Marketplace offers users various IoT services in 10 categories including asset tracking, logistics, and security [6]. Lately, Amazon features an IoT solutions platform named AWS IoT. AWS IoT offers ten different IoT services [7]. A study offers an IoT system that monitors cry detection for the babies and also RFID assisted movement detection in observing body movements of the baby and a baby room temperature detection [8]. This system alerts the parents when the baby needs attention. Another study proposes a system that collects information from different users with IoT sensors to use in agricultural services [9]. Required agricultural information is provided through the cloud to the users when needed.

Among the vast amount of services and applications, selecting the ones which will suit the users' needs best becomes a challenging issue. In this context, recommender systems provide a solution to identify and offer the services and applications which are the most appropriate to the users' preferences. In IoT environments, recommendation systems may suggest proper services based on the devices that users own [1] since the devices have specific services and applications.

The motivation of this paper comes from the need for an accurate recommendation mechanism for the service IoT users. As the technology evolves different form of IoT devices and IoT services are offered to people. Among this vast amount of available options, it is hard for users to decide which service best fits their needs. For example, when a user wants to buy a specific smartwatch, she will have numerous alternatives. If she gives some information about her previous preferences about the mobile applications she used or the services, she uses online then the number of alternatives can drop to a few. This will ease the decision process about the smartwatch. In the recommendation literature, various recommendation algorithms are used to offer new products to the users [8-10]. However, when the service IoT literature is investigated there are a few studies that offer recommendation systems for the IoT services. Few studies in the literature used tripartite graph-based model [13], artificial neural network (ANN) [14], user-based collaborative filtering [15], and a user profile similarity-based model [16]. Still, IoT services need a recommender system which offers new services to the users that best fits their interests [17]. Therefore, this paper aims to propose a system for IoT service recommendation that works with high accuracy.

This study proposes a technique to recommend services and applications for IoT users based on the devices they possess. The proposed method is the deep autoencoders which use the neural networks to predict the service preferences of the users. The innovation in this paper comes from the proposed method its application in IoT service recommendation. Deep autoencoders can use the principles of neural networks and can also perform matrix predictions. Although deep autoencoders are used in the recommendation literature before [16-19], to the best of our knowledge deep autoencoders has not been used in IoT service recommendation before. The reason for selecting deep autoencoders lies from its success in the previous recommendation applications. Deep autoencoders showed superior performance over singular value decomposition methods, user based and item-based collaborative filtering, matrix factorization and various other state-of-the-art-methods [16-20]. Deep autoencoders are expected to be successful in constructing recommendations for IoT service users. The proposed technique is compared with the state-of-the-art methods, and the superiority of deep autoencoders are demonstrated with a real-world IoT dataset. Experimental analysis is conducted to identify the best parameters for the deep autoencoders. This study is a contribution to the IoT service recommendation literature because it offers a useful tool for users' preference prediction with a new dataset that has never been used before in a recommendation study. The innovativeness of this study can be summarized with these three points:

- To the best of our knowledge, it is the first study that uses deep autoencoders to recommend IoT services and applications.
- This study proposes a real-world IoT service and application usage dataset to be used in a recommendation methodology. This dataset has not been used in a recommendation study before.
- The tool provided in this study brings more accurate results than the previously proposed state of the art recommender methods.

The remainder of the paper is as follows. The second section is related to background information about recommender systems and applications in IoT and the recommendation systems with autoencoders. The third section is about the autoencoder methodology and the real-world dataset used in this study. The fifth section explains the results of this study including comparison with the state-of-the-art methods. Results also state the effects of various parameters on the performance of deep autoencoders. The final section summarizes the study and reaches some critical conclusions.

## 2. BACKGROUND

### *2.1. Recommender Systems*

Recommender systems use content-based or collaborative filtering methods to provide recommendations for users. Content-based recommender systems use the content information of the previous items the user liked to conduct recommendations [23]. Content of a text can also be used to provide recommendation. A study proposed a sentiment analysis based on the text in the tweets [24]. Authors created a system that assess the strength of threat for a public event based on the tweets.

Collaborative filtering methods are extensively used to establish recommendations both in practice and in academia. A collaborative filtering model tries to find the anticipated items depending on the similarities between the users or the similarities between the items [25]. Collaborative filtering has a broader range of application areas because it does not require a textual definition of the items unlike content-based methods [26].

There are two types of collaborative filtering which are memory based and model-based approaches. In memory-based methods, the similarity between the users and items are calculated with similarity metrics such as Pearson correlation or Vector Cosine to find the nearest neighbors. Similarity scores between the selected user and the nearest neighbors are used to estimate future preference of the user.

In a model-based approach, a model is developed from user ratings for the items mostly using either a probability approach or rating prediction approach [27]. Clustering, classification and dimensionality reduction methods are heavily used in the modeling process [27], [28]. Studies indicate that model-based approaches have a better predictive ability when compared to memory based methods [29]. Additionally, model-based approaches handle the problem of sparsity better than memory based approaches. [28]. As a disadvantage, model-based approaches can lose information during the use of dimensionality reduction techniques [28]. Biased matrix factorization [30] and non-negative matrix factorization [31] are the most popular state-of-the-art dimensionality reduction techniques used in model-based approaches for recommender systems.

Deep learning is a specific field of machine learning and uses neural networks architecture with a high number of hidden layers. Recently deep learning has provided remarkable success on image recognition, natural language processing, and sequence prediction. Deep learning is also used for dimension reduction and provided remarkable results when compared to state-of-the-art techniques[32]. The success of deep learning grabbed the attention of recommender system appliers [22]. The first study using deep learning techniques in collaborative filtering featured restricted Boltzmann machines (RBM) [33]. Further attempts using deep learning in collaborative filtering included new neural matrix factorization model which integrates matrix factorization and multilayered perceptron under the name of Neural Collaborative filtering [34]. Another study used Long Short-Term Memory neural networks to capture dependencies between users and items in collaborative filtering [35]. Deep autoencoders predict a user-item matrix using the system of neural networks architecture. Recently there have been several successful attempts to apply autoencoders in collaborative filtering [16,18,20] which will be discussed in the upcoming sections of this paper.

### *2.2. Previous Work on IoT Recommender Systems*

The literature on IoT service recommendation is still an immature topic. The reason for that may be the difficulty of finding an appropriate dataset for an IoT recommendation model. Various websites on the Internet have IoT usage data [33]. However, usage data is not linked to the specific user, and this makes the data improper to be used for recommender systems. Still, some studies applied recommendation techniques to the IoT service data.

Mashal et al.[1] proposed a formal algorithm for IoT service recommendation. Their proposed model was a weighted undirected tripartite graph-based model with weight spreading. Mashal et al.[11] applied their previously proposed model to a real-world dataset including 110 objects, 90 services collected from Libelium, Telus, and blueRover catalogs. They reached 400 users of these services and surveyed about their preferences for the IoT services. Researchers evaluated their weighted undirected tripartite graph algorithm in comparison with some other popularly used recommendation methods. The findings demonstrated that the author's algorithm outperformed the prominent recommendation techniques.

A paper establishes a recommendation system using the connectivity of IoT devices [14]. Authors used an artificial neural network (ANN) to do the reasoning of the context and to determine the time and the product for the recommendation. Their results indicated that the ANN model produced successful recommendations regarding time and context in 98% of the incidents.

Another study offered user-based collaborative filtering with a group similarity algorithm that takes into account the membership in a group [15]. They calculated the similarity between user groups with the variables of group size, common members, and member preferences. They used similarity scores to predict service usage. The findings demonstrated that the proposed technique revealed better prediction results than the traditional group recommendation approach [15]. Another study provided a service recommendation for social IoT [37]. They proposed a recommendation metric based on time features of transactions and social relationships among devices. They applied the proposed algorithm on the simulated data and results showed that the approach outperformed the previous methods [37].

In a different study, user profile similarities are taken into account to recommend best IoT services [16]. They used a similar dataset as in [11]. The precision score of the proposed system was higher than the previously proposed methods [16]. Another study applied a characterization algorithm to build an IoT service catalog which is updated each time a user experiences a new service. Authors developed a graphical user interface for their IoT service recommender system [38].

Jeong et al. [36] proposed a method to analyze IoT device usage patterns and offered an architecture to recommend devices with a rule-based system. A rule-based segmentation system is obtained by analyzing usage patterns. The model also considered common usage patterns and similarities between users [39]. Another study is about service matchmaking which enables IoT service recommendation [40]. With service matchmaking, users can query fast and discover appropriate services. Authors propose a service matchmaking method based on Weighted-Word Latent Dirichlet Allocation (WW-LDA) which is a probabilistic topic model to extract latent semantic factors [40].

There are also studies about service recommendation in the literature. Another study uses dynamic programming and genetic algorithm to offer new web services to the users. Their results showed that the provided method is valid and has high accuracy in service recommendation [41]. A recent study proposed a scenario-based e-commerce recommendation based on customer interests and sensitivities. User-Based collaborative filtering is used to recommend products for sensitive scenarios [42]. Another study applied data mining techniques for customer choice behavior in the Internet of things, and their application predicted customer choices [43].

### *2.3. Previous Work on AutoEncoders in Collaborative Filtering*

First attempt to use autoencoders in collaborative filtering is the study of Ouyang et al. [16]. They applied autoencoders on the two of the most popular recommender system datasets which are MovieLens 100k and MovieLens 1M. They proposed different versions of autoencoders such as untrained and RBM-pretrained. Results demonstrated that RBM-pretrained autoencoders performed when compared to the nearest neighbor and singular value decomposition methods [18].

Strub and Mary [17] applied item-based and user-based denoising autoencoders on the two famous recommender datasets. Results revealed that the autoencoders demonstrated comparable results with the state-of-the-art measures. While item-based autoencoders performed best in one of the datasets, user-based autoencoders performed best in the other dataset.

Another successful attempt of applying autoencoders in collaborative filtering is found in the study of Sedhain et al. [18]. Authors employed user based and item based autoencoders in Movielens Datasets. Item-based autoencoders were found to be the best when compared with RBM, Biased MF, and user-based autoencoders. Y. Wu et al. [19] proposed a Collaborative Denoising Auto-Encoder (CDAE) model to identify top-N recommendations. Experimental results showed that CDAE demonstrates better performance than the state-of-the-art methods in estimating recommendation predictions [21]. A recent study used deep autoencoders with a new iterative output re-feeding technique [22]. Authors claim the new technique allowed to increase the learning rate and advance generalization performance of the method. Concerning rating prediction, the proposed model outperformed other popular methods. Authors also explored the effects of using different activation functions and found that exponential linear units (ELU), leaky relu (LRELU), and scaled exponential linear units (SELU) performed better results than the other activation types [22].

As can be seen from the previous work on AutoEncoders in collaborative filtering, various forms of autoencoders outperform the state-of-the-art methods in the literature. Past superior performance of autoencoders is the main reason to choose autoencoders to perform recommendations in service IoT. This study tries to find out whether autoencoders can be a successful alternative in IoT service recommendation as seen in other applications. With this aim, autoencoders are applied in a new IoT dataset and similar to the studies in the literature, findings are compared with other state-of-the-art methods. According to the results, Deep autoencoders improved performance indicators varying between 13.5% and 69.5% compared to other methods. Next section explains the methodology used in this study.

## 3. METHOD

### *3.1. Autoencoder Method*

The model in this study is inspired by the findings that autoencoders provide remarkably good results in recommender systems. Since the performance of neural networks is found to be prosperous in many kinds of the applications [44], in this study autoencoders are used to create service and application recommendations to the IoT users.

Autoencoders (AE) are neural networks that intend to reproduce their inputs to form outputs in the same shape

[45]. To produce outputs, an autoencoder implements two transformations. The first transformation encodes the inputs into a latent representation and the second transformation converts the latent representation into the output [46].

Steps of the one layered autoencoding algorithm can be summarized as follows [47]:

1) The first row of user preference data about all the items is taken as the input.
2) The taken input is encoded into a vector *h* which is a lower dimensional vector. The *f* function to find the *h* values is an activation function which can either be a sigmoid function, relu, elu, tanh or a different function. *h* is computed like this:

$$h = f(W * x + b) \qquad (1)$$

*x* is the input matrix*; w* are the weights given to the input and *b* resembles the bias.

3) Vector *h* is used to find the output or the predicted values which are recreated inputs. The output vector is created with *g* function which uses *h* values. *g* function is hyperbolic tangent function and target values are computed like this:

$$Output\ value\ (r) = g\big(f(W * x + b)\big) \qquad (2)$$

4) Error is computed as the difference between the input and the output vector. The aim is to minimize this error.
5) Backpropagation is applied from output layer to the input layer to update weights so that the error is lowered. Weights are updated with stochastic gradient descent methods that consider how much weights are responsible for the error term.
6) Step 1 to step 5 is applied for each user in the dataset.
7) Steps are repeated for more epochs. One epoch is completed when all the rows in the dataset have passed through the network.

The aim is to minimize the error which is mostly defined as the difference between the input values and the output values. Encoding and decoding steps can be extended by adding more layers. If encoding step has more than one layers, then each layer value is computed with equation 1. Similarly, in the decoding step each layer value is computed with equation 2. As layers increase, the network becomes deeper. Figure 1 demonstrates a 5-layered autoencoder.

In figure 1, $w_e^1$ is the weight matrix for the input values of x in the first layer. $e_1$ is the computed or encoded matrix in the first layer. $e_1$ is used as the input matrix for the second layer. $w_e^2$ is the weight matrix, $e_2$ is the encoded values in the second layer. *z* is the matrix version of $e_2$ values. in the first two encoding layers *f* which is sigmoid is used as the encoding function. $d_1$ is the third layer in the whole network which is also the first layer in the decoding step. *z* matrix is used as the input in the third layer, $w_d^2$ are weights for z and $d_1$ is the decoded matrix in the third layer. $d_1$ is used as the input for the fourth layer in which $w_d^1$ is the weight matrix, and the $d_2$ are the matrix for decoded target values. In the decoding step *g* function which is hyperbolic tangent is used as the decoding function. In all the layers in Figure 1, *b* represents the biases in the corresponding layers.

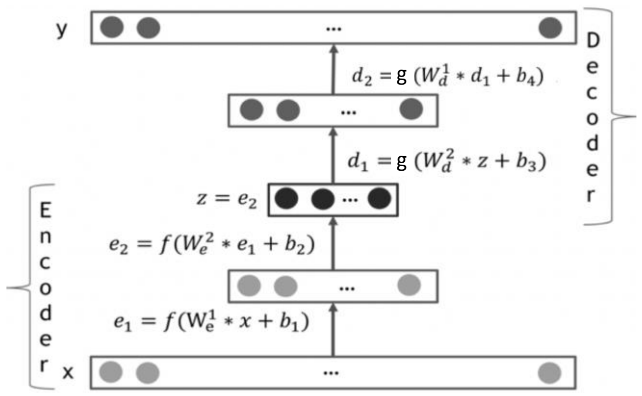
y
$d_2 = g\,(W_d^1 * d_1 + b_4)$
$d_1 = g\,(W_d^2 * z + b_3)$
$z = e_2$
$e_2 = f(W_e^2 * e_1 + b_2)$
$e_1 = f(W_e^1 * x + b_1)$
x
Decoder
Encoder

Figure 1: A five-layered AutoEncoder [20]

When noise is added to the input data during the encoding phase than the autoencoder becomes de-noising [22]. In brief, an autoencoder is a remarkable tool in dimensionality reduction with utilizing the idea in principle component analysis while strengthening its performance with the power of neural networks [48]. Since the aim in the autoencoder is to make the target values equal to the input values, it is an excellent tool for recommender systems to predict user item preference matrix from the real user-item matrix.

In this study autoencoder, with a feed-forward neural network will be implemented. In each epoch of the implementation, target values are computed with equation 2 and error is calculated with the loss function. At the end of the epoch, weights are updated to lower the error, and then the next epoch starts from the beginning. This process is called the back propagation in neural networks. The loss function in this model is Mean Squared Error (MSE). At the end of each epoch, error terms are calculated with MSE, and weights are updated depending on the error terms to obtain a better performance. Another epoch starts meaning the system runs again from the beginning with the updated weights. MSE is calculated as follows:

$$MSE = \frac{\sum_{j=1}^{n}\left(y_{ij} - r_{ij}\right)^2}{n} \qquad (3)$$

Where $y_{ij}$ is the predicted rating for the user *i* and the item *j* and the $r_{ij}$ is the real rating for the user *i* and the item *j.*

Preference prediction faces the issue of overfitting when the target values are calculated. Trading an unconstrained autoencoder with a constrained one by using a regularization term in the loss function helps this issue [22]. Therefore, in this study regularization term is added to the model and the loss function is calculated as follows:

$$C(r,y) = \frac{\sum_{j=1}^{n}(y_{ij} - r_{ij})^2}{n} + \frac{\lambda}{2} * \|w\|^2 \qquad (4)$$

Where $\lambda$ is the regularization parameter, the second term in the above equation is a regularization term. This term aims to decrease the magnitude of the weights of the inputs and help to prevent the overfitting problem [15].

### *3.2. The Internet of Things User Service Real World Dataset*

In a collaborative filtering problem, there are n users and m items producing a *nXm* preference matrix. For example, if we say $r_{ij}$ in the matrix, this resembles the rating or the preference indicator of the user $i$ for the item $j$. In this study, we produced an IoT service and application preference matrix for the users from the Social IoT Dataset published in Atzori et al.[46] and Marche et al.[15]. The first part of our methodology is to construct a user-service preference matrix from the user-device information in Social IoT Dataset.

In most collaborative filtering studies, a user item matrix is composed of user's ratings for the items like movies, jokes or other products. However, IoT based services are not tangible to rate their quality or may not be convenient to be rated each time they are accessed [15]. In this study, we used two main sections of Social IoT Dataset which are object profile and object description. From the object profile section, user identification number (id) and device type are extracted. From the object description section, services and applications offered depending on the device type are selected. Using the information from these two sections, we created a user service and application matrix. Each device type has its services and applications offered. By identifying the device types, each user possesses we determined which services the users are currently able to experience. A user may have more than one device type. Therefore, the service and application can be reached from more than one device for a user. The frequency of the object to reach the services and applications is regarded as the implicit rating of the user for the given service or application. This is based on the assumption that if a user has access to a service or applications from more than one device, the user may have a higher preference for that service or application when compared to other services. A similar assumption is found in the study of Lee and Ko [13]. They computed implicit ratings for an IoT data as the frequency of service access assuming that if a user prefers to access service more than once in a certain time, she tends to use this service more often than other services [15].

Based on the Social IoT Dataset [15, 46], created the user-item matrix has 3939 users and 46 services and applications. After the matrix creation, it was discovered that five services and nine applications are not used by any of the users in the dataset. Therefore, these services and applications were removed from the matrix. The remaining matrix included 13 services, 19 applications making a total of 32 services and applications. The smallest implicit rating is 1, and the highest implicit rating was 6. Rating of zero means there is no usage of the service by the selected users. The final version of the dataset used in this study is a 3939x32 dimensional preference matrix, in which there are 3939 users and 32 IoT services and applications. A section of the data we created from the Social IoT dataset can be seen in Table 1.

Table 1: A section from the data created from Social IoT Dataset

| User id | Service 1 | … | Service 13 | App 1 | … | App 19 |
|---|---|---|---|---|---|---|
| 2192 | 2 | … | 3 | 2 | … | 2 |
| 1433 | 1 | … | 2 | 1 | … | 1 |
| 1100 | 1 | … | 1 | 1 | … | 1 |
| 2620 | 2 | … | 3 | 2 | … | 2 |
| 3670 | 0 | … | 2 | 0 | … | 0 |
| ….. | … | … | … | … | … | … |
| 3164 | 2 | … | 3 | 2 | … | 2 |

To interpret Table 1, it can be said that user 2192 can reach service 1 from 2 devices and application 1 from 2 devices. User 1100 can reach service 1 and application 1 from 1 device.

## 4. EXPERIMENTAL EVALUATION

### *4.1. Experiment Setup*

Autoencoders model described in the previous sections is evaluated on the created IoT dataset. In the IoT dataset with 3939 users and 32 services and applications, there are 126048 implicit ratings, and 34268 of all the ratings are zero, meaning the identified users do not possess the selected services or applications.

In the dataset, 75% of the data is used for the training set while the remaining 25% is used for the testing set. Encoding dimension, regularization parameter and number of batch size in each epoch are the most prominent parameters that affect the performance of the autoencoder model. Parameters used in the model is determined by trial and error method [16,20]. The best parameters revealing the minimum error are chosen. The encoding dimension of 8 has worked best for the model by revealing the lowest loss function scores. Eight encoding dimensions mean the matrix is compressed as being eight dimensional in the encoding phase. Then the matrix become 32 dimensional again after the decoding phase. Regularization parameter λ and number of batch size are used as 0.0001 and 256 respectively.

To evaluate the prediction performance of the proposed method, mean absolute error (MAE) and root mean squared error (RMSE) metrics are used. MAE calculates the mean of absolute differences between the predicted and the real values. RMSE estimates the square root of the mean squared differences between predicted and real values. Larger deviations from the real value are penalized in RMSE more than MAE. Both of the metrics are calculated as follows:

$$RMSE = \sqrt{\frac{\sum_{j=1}^{n}\left(y_{ij} - r_{ij}\right)^2}{n}} \quad (5)$$

$$MAE = \frac{\sum_{j=1}^{n}\left|y_{ij} - r_{ij}\right|}{n} \quad (6)$$

RMSE and MAE are used to evaluate the prediction performance of the proposed models after the model has finished working and computed all the predictions. On the other hand, loss function given in equation 3 is used inside the model in the backpropagation step to calculate the error terms for updating the weights in the model.

Both autoencoder and deep autoencoder models were used in this study. In an autoencoder, there is one hidden layer while a deep autoencoder has several hidden layers. Using deep Autoencoders is an improvement for the traditional AutoEncoders model. In the deep Autoencoders model, the number of hidden layers is determined with the trial and error method [16,20]. After trying several alternatives, it is observed that a three-layer deep autoencoder model provided the lowest loss function results. When more than 3 hidden layers are added to the system, loss function results become to increase. For this reason, experiments of this study used a three-layer deep autoencoder of this study. Initially, there are 8 units in the hidden layers of both AE and deep AE. In the upcoming steps of the study effect on number of units in the hidden layers of Deep AE will also be explored.

*4.2. Experimental Results*

Firstly, the dependency of the loss function on the number of epochs for both autoencoder and deep autoencoder models are observed. Figure 2 and Figure 3 shows how the loss function changes depending on the number of epochs when the number of hidden units is 8 for both autoencoder and deep autoencoder respectively. The figures show that the models get lower loss function scores as number of epochs increase. According to Figure 2, after the number of epochs is larger than 100, the error for the autoencoder model stay relatively the same. Therefore, the number of epochs for the autoencoder model is chosen as 100 in this study. More than this number of epochs the error computed by the loss function does not change for the training and the testing set. Figure 3 shows that deep autoencoder provided faster stability as the number of epochs change. After the number of epochs is 50 deep autoencoder model provided stable error amounts for the training and the testing set. Therefore, the number of epochs for the deep autoencoder model is chosen as 50 in this study. These figures show that Deep autoencoder model requires smaller number of epochs, therefore, it has higher computing efficiency than the autoencoder models.

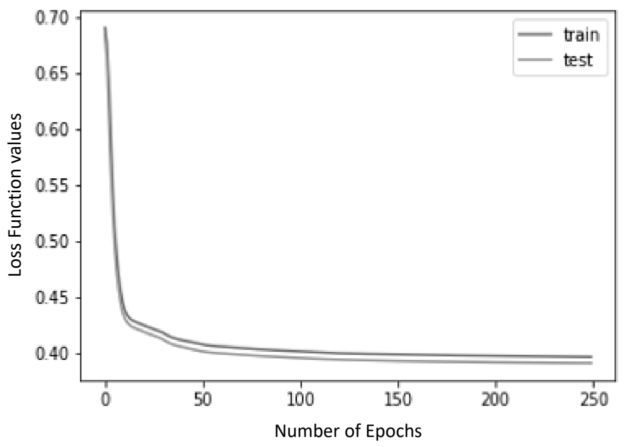

Figure 2: Loss function values depending on the number of epochs for Autoencoder Model

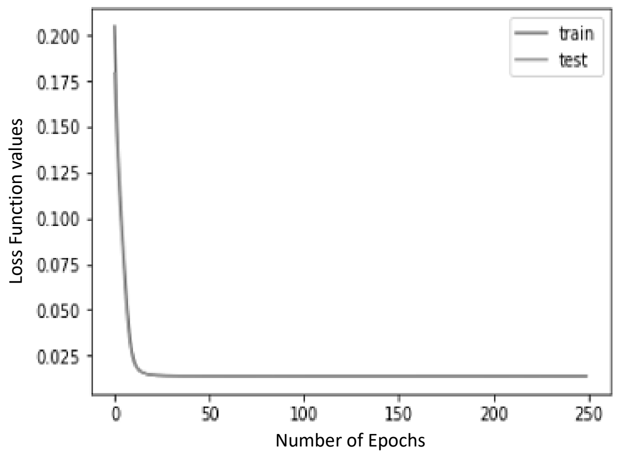

Figure 3: Loss function values depending on the number of epochs for the Deep AutoEncoder Model

*4.3. Experimental Comparisons*

In this section, we compared the proposed methods with the state-of-the-art recommendation methods used in the literature. The chosen state of the art methods performs rating prediction either based on the dimension reduction technique (Biased Matrix Factorization and Non-negative matrix factorization) neural networks (Restricted Boltzmann Machines) or pairwise similarity (item-based collaborative filtering). These methods are chosen for comparison since they are the most popular state of the art methods in their category.

The selected state of the art methods to compare the proposed autoencoder and deep autoencoder models are as follows:

**Biased Matrix Factorization (Biased MF):** Biased MF models user-item matrix in a way to an unknown latent space of $f$ dimensionality [30]. Biased MF predicts ratings by learning the interaction between the user's latent vector and the item's latent vector [50]. Steps of biased matrix factorization are as follows [47, 48]:

- A matrix R having n rows representing the number of users and m columns representing number of items contains scores of all users. This matrix will be factorized
- P and Q matrices are constructed. P matrix is n x K and Q matrix is K x m so that two matrices of which product need to be approximating R
- K resembles the number of latent features;

Biases explain some portion of this interaction. The formula for biased MF is as follows [50]:

$$r_{ij} = \mu + b_i + b_u + P * Q \tag{7}$$

Where $b_i$ and $b_u$ demonstrate the bias of user $i$ and item $j$, respectively. $\mu$ is the average rating.

**Non-negative Matrix Factorization (NMF):** NMF is a special type of MF where the values in the user-item matrix are restricted to be only the positive values [31]. Since the actual preference values in a user-item matrix are non-negative, NMF can be properly applied in prediction. NMF has the following error function [31]:

$$\underset{P,Q}{\operatorname{argmin}} \underset{R \leftrightarrow PQ}{d} = \|R - PQ\|^2 \; s.t. P, Q \geq 0 \tag{8}$$

where $R$ shows the predicted matrix.

**Restricted Boltzmann Machines (RBM):** Restricted Boltzmann Machines (RBMs) are neural networks that can be classified as energy-based models [52]. A restricted Boltzmann machine is a two-layer model in which the first layer includes observed data which are called visible units, and the second layer consists of latent variables which are called hidden units. Each layer pairwisely connect and they are restricted not to have within layer connections [53]. RBMs are probabilistic models. Instead of assigning discrete values to the ratings it assigns probabilities. Each neuron has a binary state in the RBM model as either 0 or 1 [52]. With this binary logic, RBM predicts whether the user will like or dislike a movie. For further information about RBMs and the equations behind the method, Marlin and Swersky [50] and Salakhutdinov et al. [30] can be observed.

**Item-based Collaborative Filtering:** Item-based collaborative filtering (CF) method predict a user's rating for an item by using similarities between items in the dataset [27]. To predict the score of user A to the item B, item-based CF first calculates similarity scores between items. Real ratings of the most similar items are determined. Similarity scores are regarded as weights of the ratings for prediction calculation. The weighted average of the ratings is the predicted score of user A for item B [54]. In this study, the Pearson correlation coefficient is used to calculate similarities between items. For more information about item-based rating prediction Aggarwal [51] can be observed.

Table 2 indicates the performance of autoencoder (AE) and deep autoencoder (Deep AE) models compared with the four state-of-the-art methods. Table 2 shows the MAE and RMSE values of the six tested models. Overall, AE and Deep AE outperform the remaining state of the art methods in all of the performance metrics. The MAE and RMSE of Deep AE are 0.1010 and 0.1297 respectively followed by the AE with an MAE of 0.1221 and an RMSE of 0.1297. The third best method is RBM which is another deep learning methodology. Among the MF models, Biased MF revealed the fourth best performance while NMF revealed the worst prediction errors. Deep AE reduced the prediction error of Biased MF by %69 which is a favorite dimension reduction technique in recommender systems. Percentage improvement in prediction accuracy with deep AE, when compared to the state-of-the-art methods, can be seen in Table 3. Table 3 shows that Deep AE improves the accuracy of state-of-the-art techniques with a minimum of 63.8% and a maximum of 83.1%. Deep AE also improves the prediction accuracy of a traditional AE over 10 %. These results indicate Deep AE provides superior performance when compared with the state-of-the-art methods and the traditional AE method.

Table 2: MAE and RMSE values of the compared models

| Method | MAE | RMSE |
|---|---|---|
| Biased MF | 0.3313 | 0.4118 |
| NMF | 0.6839 | 0.7693 |
| RBM | 0.2790 | 0.3844 |
| Item-based CF | 0.4775 | 0.6363 |
| Autoencoders (AE) | 0.1221 | 0.1499 |
| Deep AE | 0.1010 | 0.1297 |

Table 3: Percentage improvement in accuracy with Deep AE

| Method | % improvement in MAE | % improvement in RMSE |
|---|---|---|
| Biased MF | 69.5 | 68.5 |
| NMF | 85.2 | 83.1 |
| RBM | 63.8 | 66.3 |
| Item-based CF | 78.8 | 79.6 |
| Autoencoders (AE) | 17.3 | 13.5 |

The superior performance of Deep AE is mainly because utilizing neural networks in recommender systems lead to understanding the complex relations between the users and items better. This thorough understanding increased the accuracy of the predictions. Deep AE has less prediction error than AE. The reason for this finding is that deep AE has three hidden layers while AE has only one. Increasing the number of hidden layers in both the encoding and the decoding part of the model reduced the prediction errors and improved the performance of the AE model.

### 4.4. *Effects of the Number of Hidden Units in Deep AE*

The effect of the number of units in hidden layers of deep AE on the performance metrics is also explored. Table 4 shows how MAE and RMSE change according to the number of units in the hidden layers of Deep AE. Results in Table 4 are obtained when the Deep AE run for 100 epochs. As Table 4 demonstrates, the lowest loss values are observed when the number of hidden units are 8 and 16. For a 32-item matrix, 8 or 16 number of hidden units are found to be sufficient.

Table 4: MAE and RMSE values of Deep AE for the different number of units

| Number of hidden units | MAE | RMSE |
|---|---|---|
| 2 | 0.0586 | 0.0723 |
| 4 | 0.0601 | 0.0737 |
| 8 | 0.0570 | 0.0707 |
| 12 | 0.0601 | 0.0708 |
| 16 | 0.0596 | 0.0705 |
| 24 | 0.0912 | 0.1129 |

### *4.5. Effects of the Activation Functions*

The primary purpose of activation functions is to convert an input signal to an output signal so that the output can be used as input in the next layer [55]. To investigate the effects of using different activation functions, we examined some of the most common activation types used in neural networks which are RELU (rectified linear units), sigmoid, TANH (hyperbolic tangent), ELU (exponential linear units), and SELU (scaled exponential linear units). Table 5 shows the dependency of the performance metrics on the type of activation function for the Deep AE model that run 100 epochs with 8 number of hidden units.

As seen in Table 5, ELU has the highest performance among all the activation types. We initially used the RELU activation function to compare deep AE with other techniques. After evaluating the effects of different activation functions, it is observed that using ELU the performance of Deep AE could further be increased. The success of ELU in deep learning has been documented before. Clevert et al.[53] showed that ELU revealed faster results and generated more accurate classification scores. The success of ELU can be explained by its ability to decrease the vanishing gradient problem by giving identity to positive values [56].

Table 5: MAE and RMSE values of Deep AE for the different activation functions

| Activation Function | MAE | RMSE |
|---|---|---|
| RELU | 0.1010 | 0.1297 |
| Sigmoid | 0.0817 | 0.1018 |
| TANH | 0.0578 | 0.0715 |
| ELU | 0.0570 | 0.0707 |
| SELU | 0.0648 | 0.0796 |

### *4.6. Discussion of the Results*

AE and Deep AE generated the best performance in recommending services and applications for the IoT device users when compared to the state-of-the-art recommendation techniques. Deep AE performed better than AE due to its deeper architecture with three hidden layers. The high performance of the Deep AE can be explained neural networks' ability to model complex, high level, nonlinear relationships with multiple hidden layers [50]. This effective modeling makes the neural network learn the pattern in the data and construct accurate predictions regarding the preferences of users for the services and applications in IoT. The analyses showed that the performance of the Deep AE could even be increased by using the ELU activation function which can eliminate vanishing gradient problem.

When the effects of hidden units are analyzed, it is seen that there is not a clear pattern stating a direction when the number of hidden units is increased. Hidden units of 8 and 16 provided the best results. The reason for not observing a clear directional relationship between hidden units and the performance of deep AE can be explained by the nature of the data. IoT data used in this study has 32 dimensions meaning there were 32 services and applications in total. Since the data does not have a high number of dimensions increasing the hidden units, do not make a significant difference. Even, increasing the hidden units a lot can lower the performance as in the case where the number of hidden units is 24. When the number of hidden units is 24, the model revealed the worst prediction performance.

## 5. CONCLUSION

This study proposed autoencoder models for predicting users' preferences for IoT services and applications. As the dataset, social IoT data of Atzori et al. [46] is obtained, and the data for users' service and application possession is put in a matrix format suitable for recommender systems. The constructed IoT dataset is used to apply the proposed autoencoder and deep autoencoder models. The proposed models were compared with the three state-of-the-art methods which are Biased Matrix Factorization, Non-Negative Matrix Factorization, and Restricted Boltzmann machines. Results showed that autoencoders outperformed the three popular techniques in the selected performance metrics. Deep autoencoder also exceeded the autoencoder method's performance due to its three hidden layers. Findings also showed that changing the activation function will increase the prediction performance of the deep autoencoder model even more.

This study is an essential contribution to the IoT literature because it proposed an effective tool to recommend services and applications for IoT users. The accuracy of the deep autoencoder's predictions is evidence that this tool can offer new services and applications which the IoT users are more likely to favor. The superiority of the results also confirms that the deep autoencoder method is a valid tool to perform IoT service recommendation. In the future, the author plans to apply the deep autoencoders on more complex, higher dimensional real IoT datasets. The other future goal of this study is to propose an autoencoder model in which real-time IoT data is used and the recommendations are updated as the amount of data is increasing.

## REFERENCES

[1] I. Mashal, T. Y. Chung, and O. Alsaryrah, "Toward service recommendation in Internet of Things", **International Conference on Ubiquitous and Future Networks**, *ICUFN*, 328–331, 2015.

[2] Govloop Community, "The Internet of Things: What the IoT means for the public sector", *Isaca*, 6, 2013.

[3] E. Ahmed *et al.*, "The role of big data analytics in Internet of Things", *Computer Networks*, 129, 459–471, 2017.

[4] D. Çulha, "Blockchain of Meetings of IoT Devices", *Bilişim Teknolojileri Dergisi*, 14(2), 129–136, 2021.

[5] I. Şafak, E. Ünsal, "Türkiye'de Dağıtık Hesap Defteri Teknolojili Nesnelerin İnterneti Ödeme Sistemleri için Sistem Tasarım Önerileri", *Bilişim Teknolojileri Dergisi*, 23–36, 2021,

[6] Telus, "IoT Marketplace", *Telus*, 2018.

[7] İnternet: Amazon Web Services (AWS), https://docs.aws.amazon.com/en_us/aws-technical content/latest/aws-overview/internet-of-things-services.html, 26.12.2018.

[8] Srinivasa K G, Sowmya BJ, A. Shikhar, R. Utkarsha, A. Singh, "Data Analytics Assisted Internet of Things Towards Building Intelligent Healthcare Monitoring Systems", *Journal of Organizational and End User Computing*, 30(4), 83–103, 2018.

[9] S. S. Gill, I. Chana, R. Buyya, "IoT Based Agriculture as a Cloud and Big Data Service", *Journal of Organizational and End User Computing*, 29(4), 1–23, 2017.

[10] A. Agarwal, M. Chauhan, "Similarity Measures used in Recommender Systems: A Study", *International Journal of Engineering Technology Science and Research*, 4(6), 2394–3386, 2017.

[11] B. Zhang, B. Yuan, "Improved collaborative filtering recommendation algorithm of similarity measure", *AIP Conference Proceedings*, 1839, 2017.

[12] Y. Wang, J. Deng, J. Gao, and P. Zhang, "A hybrid user similarity model for collaborative filtering", *Information Sciences*, 418, 102–118, 2017.

[13] I. Mashal, O. Alsaryrah, and T. Y. Chung, "Performance evaluation of recommendation algorithms on Internet of Things services", *Physica A: Statistical Mechanics and its Applications*, 451, 646–656, 2016.

[14] Y. Salman, A. Abu-Issa, I. Tumar, and Y. Hassouneh, "A proactive multi-type context-aware recommender system in the environment of Internet of Things", **15th IEEE International Conference on Computer and Information Technology**, *CIT 2015*, 2015, 351–355.

[15] J. S. Lee and I. Y. Ko, "Service recommendation for user groups in internet of things environments using member organization-based group similarity measures", **IEEE International Conference on Web Services**, *ICWS 2016*, 276–283, 2016.

[16] S. Forouzandeh *et al.*, "Recommender system for Users of Internet of Things (IOT)", *IJCSNS International Journal of Computer Science and Network Security*, 17(8), 46–51, 2017.

[17] C. Marche, L. Atzori, M. Nitti, "A Dataset for Performance Analysis of the Social Internet of Things", **IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)**, 1-5, 2018.

[18] Y. Ouyang, W. Liu, W. Rong, Z. Xiong, "Autoencoder-Based Collaborative Filtering", **International Conference on Neural Information Processing**, 284–291, 2014.

[19] F. Strub and J. Mary, "Collaborative Filtering with Stacked Denoising AutoEncoders and Sparse Inputs", **NIPS Workshop on Machine Learning for eCommerce**, 2015.

[20] S. Sedhain, A. K. Menon, S. Sanner, L. Xie, "AutoRec: Autoencoders Meet Collaborative Filtering", **24th International Conference on World Wide Web**, 111–112, 2015.

[21] Y. Wu, C. DuBois, A. X. Zheng, M. Ester, "Collaborative Denoising Auto-Encoders for Top-N Recommender Systems", **9th ACM International Conference on Web Search and Data Mining - WSDM '16**, 153-162, 2016.

[22] O. Kuchaiev B. Ginsburg, "Training Deep AutoEncoders for Collaborative Filtering", *arXiv preprint arXiv:1708.01715*, 2017.

[23] P. Lops, M. De Gemmis, G. Semeraro, "Content-based Recommender Systems: State of the Art and Trends", ***Recommender Systems Handbook***, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Boston: Springer, 2011.

[24] V. Subramaniyaswamy, R. Logesh, M. Abejith, S. Umasankar, A. Umamakeswari, "Sentiment Analysis of Tweets for Estimating Criticality and Security of Events", *Journal of Organizational and End User Computing*, 29(4), 51–71, 2017.

[25] W. W. Cohen W. Fan, "Web-collaborative filtering: recommending music by crawling the Web", *Computer Networks*, 33(1), 685–698, 2000.

[26] K. Miyahara M. J. Pazzani, "Collaborative Filtering with the Simple Bayesian Classifier", *IPSJ Journal*, 43(11), 679–689, 2002.

[27] B. Sarwar, G. Karypis, J. Konstan, J. Reidl, "Item-based collaborative filtering recommendation algorithms", **10th International Conference on World Wide Web - WWW '01**, 285–295, 2001.

[28] X. Su T. M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques", *Advances in Artificial Intelligence*, 1–19, 2009, doi: 10.1155/2009/421425.

[29] P. H. Aditya, I. Budi, and Q. Munajat, "A comparative analysis of memory-based and model-based collaborative filtering on the implementation of recommender system for E-commerce in Indonesia: A case study PT X", **nternational Conference on Advanced Computer Science and Information Systems**, *ICACSIS*, 303–308, 2016.

[30] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Tecniques for Recommender Systems", *Computer*, 8, 30–37, 2009.

[31] X. Luo, M. Zhou, Y. Xia, and Q. Zhu, "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems", *IEEE Transactions on Industrial Informatics*, 10(2), 1273–1284, 2014.

[32] O. Kaynar, Z. Aydın, and Y. Görmez, "Sentiment Analizinde Öznitelik Düşürme Yöntemlerinin Oto Kodlayıcılı Derin Öğrenme Makinaları ile Karşılaştırılması", *Bilişim Teknolojileri Dergisi*, 319–326, 2017.

[33] R. Salakhutdinov, A. Mnih, and G. Hinton, “Restricted Boltzmann Machines for Collaborative Filtering”, **24th International Conference on Machine Learning**, 2007, 791–798.

[34] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural Collaborative Filtering”, **International World Wide Web Conference**, 173–182, 2017.

[35] C.-Y. Wu, A. Ahmed, A. Beutel, A. J. Smola, and H. Jing, “Recurrent Recommender Networks**”, 10th ACM International Conference on Web Search and Data Mining - WSDM ’17**, 495–503, 2017.

[36] İnternet: V. Granville, “Great IoT, Sensor and other Data Sets Repositories”, Data Science Central, https://www.datasciencecentral.com/profiles/blogs/great-sensor-datasets-to-prepare-your-next-career-move-in-iot-int, 10.12.2018.

[37] Z. Chen, R. Ling, C.-M. Huang, and X. Zhu, “A scheme of access service recommendation for the Social Internet of Things”, *Journal of Communication Systems*, 29, 694–706, 2016.

[38] L. Noirie, M. Le Pallec, and N. Ammar, “Towards automated IoT service recommendation”, **20th Conference on Innovations in Clouds, Internet and Networks**, **ICIN 2017**, 103–106, 2017.

[39] H. Jeong, B. Park, M. Park, K. B. Kim, and K. Choi, “Big data and rule-based recommendation system in Internet of Things”, *Cluster Computing*, 1–10, 2017.

[40] Y. Liu, T. Zhu, Y. Jiang, and X. Liu, “Service matchmaking for Internet of Things based on probabilistic topic model”, *Future Generation Computer Systems*, 94, 272–281, 2019.

[41] B. Hu, Z. Zhou, and Z. Cheng, “Web Services Recommendation Leveraging Semantic Similarity Computing”, *Procedia Computer Science*, 129, 35–44, 2018.

[42] X. Wu, L. Zhang, S. Tian, and L. Wu, “Scenario based e-commerce recommendation algorithm based on customer interest in Internet of things environment”, *Electronic Commerce Research*, 0123456789, 2019.

[43] Y. Yan, C. Huang, Q. Wang, and B. Hu, “Data mining of customer choice behavior in internet of things within relationship network”, *International Journal of Information Management, Article in Press*, 2018.

[44] Ş. Birim and A. Tümtürk, “Modeling and Forecasting Turkey’ s Electricity Consumption by using Artificial Neural Network”, *American Scientific Research Journal for Engineering, Technology, and Sciences*, 25(1), 192–208, 2016.

[45] İnternet: N. Hubens, Deep inside: Autoencoders, Towards Data Science, https://towardsdatascience.com/deep-inside-autoencoders-7e41f319999f, 10.12.2018

[46] G. E. Hinton and R. S. Zemel, “Autoencoders, Minimum Description Length and Helmholtz Free Energy”, *Advances in neural information processing systems*, 3–10, 1994.

[47] İnternet: R. Khandelwal, Deep Learning Autoencoders, https://medium.com/datadriveninvestor/deep-learning-autoencoders-db265359943e, 22.02.2019.

[48] G. E. Hinton and R. R. Salakhutdinov, “Reducing the Dimensionality of Data with Neural Networks”, *Science*, 313, 504–507, 2006.

[49] L. Atzori, M. Nitti, C. Marche, “Social Internet of Things IoT Network Dataset”, *Social IoT*, 2018.

[50] N. Liang, H.-T. Zheng, J.-Y. Chen, A. Sangaiah, C.-Z. Zhao, “TRSDL: Tag-Aware Recommender System Based on Deep Learning–Intelligent Computing Systems”, *Applied Sciences*, 8(5), 799, 2018.

[51] S. Chen, Y. Peng, “Matrix factorization for recommendation with explicit and implicit feedback”, *Knowledge-Based Systems*, 158, 109–117, 2018.

[52] A. Oppermann, “Deep Learning meets Physics : Restricted Boltzmann Machines Part I”, *Towards Data Science*, 2018.

[53] B. Marlin, K. Swersky, “Inductive principles for restricted Boltzmann machine learning,” **International Conference on Artificial Intelligence and Statistics (AISTATS)**, 2010, 9, 305–306.

[54] C. C. Aggarwal, “Neighborhood-Based Collaborative Filtering”, *Recommender Systems*, 2016, 29–70.

[55] A. Singh, W. Follow, “Activation functions and it’s types-Which is better? ”, *Towards Data Scienceards*, 2017.

[56] D.-A. Clevert, T. Unterthiner, S. Hochreiter, “Fast and Accurate Deep Network Learning by Exponential Linear Units”, *arXiv preprint arXiv:1511.07289*, 2015.