Hybrid Recommendation Algorithm Based on Restricted Boltzmann Machine and Weighted Slope One: A Postprint

Authors: Shen Xueli, He Chenhao, Meng Xiangfu

Date: 2019-01-03T00:00:00+00:00

Abstract

To address the issues of sparsity and low prediction accuracy in traditional collaborative filtering algorithms, this paper proposes a hybrid recommendation algorithm based on Restricted Boltzmann Machines and weighted Slope One. Initially, the data sparsity problem is mitigated through preliminary imputation of the rating matrix using Restricted Boltzmann Machines. Subsequently, item attribute information is incorporated via a hybrid item similarity computation method. Finally, recommendation performance is enhanced through secondary prediction employing the weighted Slope One algorithm. Experimental results on the MovieLens100K dataset demonstrate that the combination of these two algorithms improves recommendation accuracy.

Full Text

Preamble

Vol. 37 No. 3
Application Research of Computers
ChinaXiv Partner Journal

Research on Hybrid Recommendation Algorithm of Restricted Boltzmann Machine and Weighted Slope One

Shen Xueli, He Chenhao, Meng Xiangfu
(School of Electronic & Information Engineering, Liaoning Technical University, Huludao, Liaoning 125105, China)

Abstract: To address the sparsity and low prediction accuracy issues faced by traditional collaborative filtering algorithms, this paper proposes a hybrid recommendation algorithm based on Restricted Boltzmann Machine (RBM) and

weighted Slope One. First, the rating matrix is preliminarily filled through RBM to alleviate data sparsity. Then, a hybrid item similarity calculation method is introduced to incorporate item attribute information. Finally, a second prediction through the weighted Slope One algorithm enhances recommendation effectiveness. Experiments on the MovieLens100K dataset demonstrate that the combination of these two algorithms improves recommendation accuracy.

Keywords: restricted Boltzmann machine; weighted slope one; adjusted cosine similarity; jaccard similarity

0 Introduction

With the rapid development of Internet technology and applications, humanity has entered the era of big data. Faced with increasingly complex data volumes, how to filter data has become a common challenge for both network service providers and Internet users. Personalized recommendation systems represent a primary approach to data filtering. Today, the most widely applied personalized recommendation method is collaborative filtering algorithms [**?**], and hybrid recommendation algorithms that combine two collaborative filtering approaches have become an important research direction in this field [**?**].

Collaborative filtering algorithms suffer from the data sparsity problem. In practical applications, since users only rate a small number of items, the user-item rating matrix is often extremely sparse, leading to low recommendation accuracy. Hybrid recommendation algorithms that combine two algorithms can effectively mitigate this issue. The Slope One algorithm is a common user-based collaborative filtering algorithm characterized by its simplicity and efficiency. It fully considers the relationships between users and items within the rating matrix and can achieve better recommendation results than other traditional collaborative filtering algorithms when data is dense [**?**]. However, its performance is unsatisfactory when data is sparse. To address this problem, researchers have proposed using methods such as Singular Value Decomposition (SVD) [**?**] [**?**] to fill missing values in the rating matrix before making predictions with the Slope One algorithm. With the application of neural network models in collaborative filtering, these models can better fit data and deeply mine relationships within it compared to traditional collaborative filtering algorithms. The Restricted Boltzmann Machine (RBM) is a neural network model that has been successfully applied in recommendation systems. Experiments have proven that RBM models achieve better recommendation performance than SVD and other models [**?**]. Therefore, this paper adopts RBM to fill the rating matrix. Additionally, researchers have introduced similarity relationships into the Slope One algorithm [**?**, **?**], proposing the weighted Slope One algorithm, which effectively solves the cold start problem and improves prediction accuracy.

In summary, this paper combines and improves upon these two models by proposing a hybrid recommendation algorithm based on an item-based

real-valued Restricted Boltzmann Machine and weighted Slope One (IR-RBM-WSO).

1 Research Status of Restricted Boltzmann Machines

Assume a recommendation system contains m users and n items, with user ratings for items being integers between 1 and k. This establishes a user-item rating matrix $R_{m\times n}$. In the item attribute information, there are s item categorical attributes. For example, the MovieLens dataset provides 18 categories of movie information. Each item belongs to one or more categories, with membership indicated by 1 and non-membership by 0. This establishes an item-attribute matrix $I_{n\times s}$. The following algorithm explanation will be based on this example.

The Restricted Boltzmann Machine (RBM) is a two-layer neural network model: the visible layer represents rating data, and the hidden layer acts as a feature extractor to capture relationships within the data [**?**]. Its characteristic is no connections within layers and full connections between layers. In 2007, Salakhutdinov et al. [**?**] first applied RBM to recommendation systems. For the integer rating range of 1 to k in recommendation systems, they changed the visible units in the original RBM model from binary variables to softmax units containing k binary units. In this model, each rating required k units to represent, making the model parameters overly complex and training time-consuming. In 2013, Georgiev et al. [**?**] proposed an RBM model based on both users and items that directly applied rating values to the visible layer. This model reduced algorithm complexity but had poor interpretability. In 2015, He Jieyue et al. [**?**] introduced social information into the model based on real-valued RBM, alleviating the data sparsity problem of RBM and improving recommendation effectiveness. However, the introduced social information had limited applicability, and the sparsity issue of social information itself made the recommendation effect less than ideal. In 2017, Huo Shuhua [**?**] proposed an item-based RBM model, which achieved better recommendation results than user-based RBM models.

Building upon the real-valued RBM model, this paper establishes a Restricted Boltzmann Machine model from an item perspective, proposing an item-based real-valued Restricted Boltzmann Machine model (IR_RBM). Additionally, we incorporate item attribute information, which is more universally present in recommendation systems, into the similarity calculation process to improve prediction results.

2 Proposed Algorithm

This paper proposes an item-based real-valued Restricted Boltzmann Machine model (IR_RBM), with its structure shown in Figure 1 [Figure 1: see original paper]. For each item, an IR-RBM model is established. In each model' s visible layer, there are m numerical units representing each user' s rating for that item. Visible layer node values are integers between 0 and k, where 0 indicates that

the user has not rated the item, and existing ratings are represented by real values.

Since real values are used directly to represent ratings, a quadratic term is added to the original RBM energy model. The energy calculation formula for this model is:

$$E(\mathbf{v}, \mathbf{h}|\theta) = -\sum_{i=1}^{m}\sum_{j=1}^{n} W_{ij} v_i h_j - \sum_{i=1}^{m} c_i v_i - \sum_{j=1}^{n} b_j h_j + \frac{1}{2}\sum_{i=1}^{m} v_i^2$$

where b_j is the bias of the j-th hidden node with an initial value of 0 [**?**]; a_i is the bias of the i-th visible unit; and W_{ij} is the connection weight between the i-th visible unit and the j-th hidden unit, initialized as random numbers following a normal distribution $N(0, 0.01)$.

Based on the energy formula, when rating data is input into the model, the activation probability of the j-th hidden unit is:

$$p(h_j = 1|\mathbf{v}, \theta) = \sigma\left(b_j + \sum_{i=1}^{m} v_i W_{ij}\right)$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the activation function. When the state of hidden units is determined, the value of the i-th visible unit is:

$$v_i = c_i + \sum_{j=1}^{n} h_j W_{ij}$$

Parameter updates use the contrastive divergence method, with the parameter update criterion being:

$$\begin{aligned}
\Delta W_{ij} &= \lambda(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{recon}) \\
\Delta a_i &= \lambda(\langle v_i \rangle_{data} - \langle v_i \rangle_{recon}) \\
\Delta b_j &= \lambda(\langle h_j \rangle_{data} - \langle h_j \rangle_{recon})
\end{aligned}$$

where λ is the learning rate. In our experiments, the Adam method is used to update the learning rate to accelerate model convergence, with an initial value set to 0.001 [**?**].

The pseudo-code for establishing the IR-RBM model is shown in Algorithm 1.

Algorithm 1: Pseudo-code for the IR-RBM Model

Input: Rating matrix $R_{m\times n}$

Output: Rating matrix filled using the IR-RBM model

1. Establish the IR-RBM model
2. for $t = 1$ to iteration do: // iteration is the IR-RBM iteration count, set to 20 here
3. for $i = 1$ to m do:
4. `// Update parameters using equations (1) to (4)`
5. end for
6. end for
7. for each $R_{ij} = 0$ in $R_{m \times n}$ do:
8. // Use trained IR-RBM to predict score for R_{ij}
9. end for

2.1 Similarity Algorithm

The composite similarity calculation method proposed in this paper first computes item similarity in the filled rating matrix using the adjusted cosine similarity method. Although this method can effectively mine item similarity relationships in the rating matrix, it ignores the similarity relationships between item attributes themselves. In practical applications, users often have similar preferences for items of the same type and give similar ratings. Second, based on this observation, this paper proposes a Jaccard similarity calculation method incorporating information entropy to mine item attribute similarity relationships. Information entropy assigns different weights to different item attributes, and then the Jaccard algorithm is used to calculate item attribute similarity. Finally, both item rating similarity and item attribute similarity are combined as the final item similarity, which serves as the weight for the Slope One algorithm.

2.1.1 Adjusted Cosine Similarity Algorithm Adjusted cosine similarity builds upon cosine similarity by incorporating a de-meaning approach to correct for different user rating scales. The item-based adjusted cosine similarity calculation formula is shown in equation (5):

$$sim(i,j) = \frac{\sum_{u \in U_{ij}} (R_{ui} - \bar{R}_i)(R_{uj} - \bar{R}_j)}{\sqrt{\sum_{u \in U_{ij}} (R_{ui} - \bar{R}_i)^2} \sqrt{\sum_{u \in U_{ij}} (R_{uj} - \bar{R}_j)^2}}$$

where R_{ui} represents user u' s rating for item i; $\bar{R}_i$ and $\bar{R}_j$ represent the average ratings for items i and j respectively.

The value range of adjusted cosine similarity is $[-1, 1]$. Since this similarity will be used as a weight in this paper, it needs to be mapped to the $[0, 1]$ interval. The mapping formula is shown in equation (6):

$$sim'(i,j) = \frac{1 + sim(i,j)}{2}$$

2.1.2 Improved Jaccard Similarity Algorithm with Information Entropy Jaccard similarity is a method for describing similarity between two sets, suitable for calculating highly discrete dimensional feature vectors [**?**]. The calculation formula is:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

where A and B are two n-dimensional vectors with values of 1 or 0 indicating attribute inclusion. The numerator represents the number of dimensions with identical values in A and B, while the denominator represents the number of dimensions in the union of A and B.

Direct application to recommendation systems leads to inaccurate similarity measurement. For a single dimension, let a_i and b_i be the values of the i-th dimension in A and B respectively. The condition $a_i = b_i$ holds in two cases: both a_i and b_i are 1, or both are 0. Jaccard similarity considers $a_i = b_i = 0$ as a similar case. However, in recommendation systems, the practical meaning of $a_i = b_i = 0$ is that neither item A nor B belongs to type i, which cannot serve as a basis for similarity. Moreover, in actual datasets, an item only belongs to a few of many categorical attributes, meaning the case where $a_i = b_i = 0$ occurs frequently. For example, assuming items are categorized by 6 attributes, where item A only belongs to attribute 1 with attribute vector $\{1,0,0,0,0,0\}$, and item B only belongs to attribute 2 with attribute vector $\{0,1,0,0,0,0\}$, the Jaccard coefficient calculates a similarity of 66.7%, while in practical terms, they belong to different types and could be considered to have a similarity of 0.

To address this issue, this paper modifies the Jaccard similarity algorithm. For a single dimension, there are four possible cases: - p: both a_i and b_i are 1 - q: a_i is 1 and b_i is 0 - r: a_i is 0 and b_i is 1 - s: both a_i and b_i are 0

To make similarity calculation more realistic, this algorithm only considers p as a similar case. The similarity between A and B is then:

$$sim_{Jaccard}(A,B) = \frac{p}{p+q+r}$$

where p represents the number of dimensions satisfying this condition.

Additionally, in Jaccard similarity calculation, each attribute contributes equally, ignoring the discriminative power of different attributes. For instance, if most (or very few) items contain attribute i, then attribute i has low discriminative power for items and its weight should be relatively reduced. To address this, this paper introduces information entropy to differentiate

attributes. Information entropy can quantify the disorder of a system. For binary values of 0 and 1, the greater the difference between the counts of 0 and 1 units, the smaller the uncertainty and the lower the entropy value; conversely, the smaller the difference, the more stable the system and the greater the entropy value.

For each column of the user-attribute matrix, the probability of value 1 appearing is calculated, and the information entropy weight of this item is computed through the information entropy calculation formula (8):

$$H(\alpha_i) = -p(\alpha_i)\log(p(\alpha_i)) - (1-p(\alpha_i))\log(1-p(\alpha_i))$$

where $p(\alpha_i)$ is the probability that α_i has value 1. Subsequently, the similarity between two items is calculated using the information entropy weights.

In summary, the improved Jaccard similarity calculation method with information entropy proposed in this paper is shown in equation (9):

$$sim_{item}(i,j) = \sum_{\alpha \in P} H(\alpha) \cdot sim_{Jaccard}(A,B)$$

2.1.3 Item Comprehensive Similarity Algorithm This paper uses the average of the two similarity measures as the final item similarity, calculated using formula (10):

$$sim_{item}(i,j) = \frac{sim_{cosine}(i,j) + sim_{Jaccard}(i,j)}{2}$$

Algorithm 2: Pseudo-code for Item Comprehensive Similarity

Input: Filled rating matrix $R_{m\times n}$, item attribute matrix $I_{n\times s}$, item set V, user set M, attribute set P

Output: Item similarity matrix $S_{n\times n}$

1. for each attribute i in P do:
2. // Calculate information entropy p_i using equation (8)
3. end for
4. for each item pair i, j in V $(i \neq j)$ do:
5. // Calculate adjusted cosine similarity $sim_{cosine}(i,j)$ using equation (5)
6. // Calculate improved Jaccard similarity $sim_{Jaccard}(i,j)$ using equation (9)
7. // Calculate comprehensive similarity $sim_{item}(i,j)$ using equation (10)
8. // In matrix S, update $S_{ij} = S_{ji} = sim_{item}(i,j)$
9. end for

2.2 Weighted Slope One Algorithm

This paper uses the weighted Slope One algorithm to calculate final prediction results. The Slope One algorithm is a recommendation algorithm based on linear regression, consisting of deviation calculation and prediction formulas. The item-based deviation calculation formula is:

$$dev_{ij} = \frac{\sum_{u \in S_{ij}} (R_{ui} - R_{uj})}{|S_{ij}|}$$

where R_{ui} represents user i' s rating for item u, and S_{ij} represents the set of users who rated both items i and j. Since this paper uses a completely filled rating matrix, S_{ij} here equals the user set M.

The Slope One prediction formula incorporating item similarity is shown in equation (12):

$$pre(u,i) = \frac{\sum_{j \in S(u)} (R_{uj} + dev_{ij}) \times sim_{item}(i,j)}{\sum_{j \in S(u)} sim_{item}(i,j)}$$

where $S(u)$ represents the N items with the highest similarity to item u selected based on item similarity.

Algorithm 3: Slope One Prediction Pseudo-code

Input: Rating matrix $R_{m \times n}$, item similarity matrix $S_{n \times n}$, user set M, defined maximum neighbor count k, item to be rated R_{ui}

Output: Predicted rating $pre(u,i)$

1. Take column vector S_u from S, select k items with highest similarity to form neighbor set $S(u)$
2. for each item u in $S(u)$ do:
3. for each user j in M $(j \neq i)$ do:
4. `// Calculate $dev_{ij}$ using equations (1) to (11)`
5. end for
6. // Calculate $pre(u,i)$ using equations (1) to (12)
7. end for

3 Experiments

3.1 Dataset and Evaluation Metrics

This section demonstrates the accuracy of the proposed algorithm and the impact of model parameters on experimental results through experiments. The

experiments use user-item rating information and item-attribute information from the MovieLens100K dataset for validation. The user-item rating information is stored in the u.data file, containing 100,000 rating records from 943 users on 1,682 movies. The item-attribute information is stored in the u.item file, containing attribute information for 1,682 movies. The experiments extract 18 movie category information fields to form the item-attribute matrix. From the user-item ratings, 20% of the rating data is extracted as the test dataset.

The experiments use Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) as evaluation metrics. MAE is defined in equation (13):

$$MAE = \frac{\sum_{i,j} |R_{ij} - \hat{R}_{ij}|}{N}$$

RMSE is defined in equation (14):

$$RMSE = \sqrt{\frac{\sum_{i,j} (R_{ij} - \hat{R}_{ij})^2}{N}}$$

3.2 Experimental Results

3.2.1 Hidden Unit Count Experiment in IR_RBM Model This experiment establishes the IR-RBM model to predict the test set, exploring the impact of hidden unit count on matrix filling effectiveness and determining an optimal number. In the IR-RBM model, training iterations are set to 80. The experimental results are shown in Figure 2 [Figure 2: see original paper].

Figure 2: MAE Values of IR-RBM Model with Different Numbers of Hidden Units

The figure shows that as the number of hidden units increases, the MAE value of the IR-RBM model first decreases and then increases, achieving optimal recommendation performance when the hidden unit count is 100. When the hidden unit count exceeds 100, recommendation performance declines due to overfitting caused by an excessively large number of hidden units serving as feature extractors. Therefore, in subsequent experiments, the hidden unit count for the IR-RBM model is fixed at 100.

3.2.2 Training Iteration Count Experiment in IR-RBM Model This experiment explores the impact of the iteration parameter on matrix filling effectiveness in the IR-RBM model and determines an optimal iteration value. The experimental results are shown in Figure 3 [Figure 3: see original paper], where the x-axis represents training iterations and the y-axis represents MAE values. The figure reveals that the model converges when iteration exceeds 20, achieving optimal performance at 80 iterations.

Figure 3: MAE Values of IR-RBM Model Under Different Training Iterations

3.2.3 Impact of Nearest Neighbor Count k on Recommendation Effect This experiment establishes the proposed IR-RBM-WSO algorithm to study the impact of nearest neighbor count K on recommendation accuracy. The experimental results are shown in Figure 4 [Figure 4: see original paper]. The model' s recommendation performance shows a trend of first improving and then stabilizing as the neighbor count increases. Since the recommendation accuracy stabilizes when $k > 80$, this paper adopts $k = 80$ as the experimental parameter.

Figure 4: MAE Values of IR-RBM-WSO Algorithm Under Different Neighbor Counts

3.2.4 Comparison Experiment with Related Algorithms Under the conditions of training iterations = 80 and neighbor count $k = 80$, this experiment compares the performance of the proposed IR-RBM-WSO algorithm with its component algorithms. The following four models are selected for comparison, with their MAE and RMSE values measured:

a) Slope One algorithm

b) Weighted Slope One algorithm based on the similarity measure proposed in this paper (WSO)

c) Item-based Restricted Boltzmann Machine model (I-RBM)

d) Algorithm using SVD for matrix filling and the weighted Slope One algorithm proposed in this paper for secondary prediction (SVD-WSO)

The experimental results are shown in Figures 5 and 6.

Figure 5: MAE Comparison of Related Algorithms
Figure 6: RMSE Comparison of Related Algorithms

Analysis of the data in both figures yields the following conclusions: The weighted Slope One algorithm outperforms the original Slope One algorithm in recommendation accuracy, with MAE and RMSE values improved by 4.8% and 8.3% respectively, demonstrating that the introduction of the proposed similarity measure alleviates the data sparsity problem of Slope One to some extent and improves recommendation effectiveness. The IR-RBM-WSO model outperforms its component algorithms I-RBM and WSO, with MAE values improved by 7.49% and 10.71% respectively, and RMSE values improved by 8.75% and 12.63% respectively, indicating that combining the two algorithms effectively compensates for their respective deficiencies and improves prediction accuracy. IR-RBM-WSO outperforms SVD-WSO, with MAE and RMSE values improved by 3.89% and 4.19% respectively, demonstrating that the IR-RBM

model proposed in this paper achieves better matrix filling performance than SVD.

3.2.5 Comparison Experiment with Existing Algorithms To further validate the performance of the proposed algorithm, this experiment compares it with the UI-RBM model proposed in literature [**?**] and the IC-CRBMF model proposed in literature [**?**] that incorporates attribute information layers based on users and items. The experimental results are shown in Figure 7 [Figure 7: see original paper].

Figure 7: MAE Comparison with Other Algorithms

The experiments demonstrate that on this dataset, the IR-RBM-WSO model outperforms the IC-CRBMF model, with the best MAE value improved by 3.8%. The UI-RBM model converges slowly and does not reach optimal performance in the comparison figure, showing weaker recommendation effectiveness than the proposed algorithm. At iteration = 500, the MAE value converges to 0.690, only 0.002 lower than the proposed algorithm. However, due to its high training iteration requirement and complex modeling of both users and items simultaneously, the training time of this algorithm is much longer than that of the proposed algorithm.

4 Conclusion

This paper proposes a hybrid recommendation algorithm that organically combines Restricted Boltzmann Machine and weighted Slope One. By filling missing values in the matrix through RBM, the matrix sparsity problem is alleviated. Through secondary prediction using the Slope One algorithm, the model' s recommendation effectiveness is enhanced. Additionally, a composite similarity calculation method is proposed to mine item attribute similarity and combine it with rating similarity, better reflecting item similarity relationships. Comparative experiments demonstrate that the proposed hybrid recommendation algorithm achieves good recommendation results. However, this algorithm cannot effectively solve the cold start problem, and improvement in this aspect will be the focus of future work. Additionally, deep learning models will be explored [**?**, **?**] to further enhance recommendation effectiveness.

References

[1] Duan M. Collaborative filtering recommendation algorithm [J]. International Journal of Security & Its Applications, 2015, 9(7): 261-268.

[2] Song Ruiping. Research of hybrid recommendation algorithm [D]. Lanzhou: Lanzhou University, 2014.

[3] Yehuda K, Robert B, Chris V. Matrix factorization techniques for recommender systems [J]. IEEE Computer Society, 2009, 42(42): 30-37.

[4] Jing Huijuan, Liang Anchun, Lin Shoude, et al. A transfer probabilistic collective factorization model to handle sparse data in collaborative filtering [C]//Proc of IEEE International Conference on Data Mining. 2015: 250-259.

[5] Xiang Xiaodong, Qiu Zixiang. Research on collaborative filtering algorithm based on slope one algorithm to improve score matrix filling [J]. Application Research of Computers, 2019, 36(5): 1-5.

[6] Du Qian. Research on the weighted Slope One recommendation technology based on clustering [D]. Beijing: Beijing University of Technology, 2016.

[7] Salakhutdinov R, Mnih A, Hinton G. Restricted Boltzmann machines for collaborative filtering [C]//Proc of the 24th International Conference on Machine Learning. 2007: 791-798.

[8] Song Shiyu, Wu Kejia. A creative personalized recommendation algorithm-user-based Slope One algorithm [C]//Proc of International Conference on Systems and Informatics. Yantai: IEEE Press. 2012: 2055-2059.

[9] Xun Sisi. Integrating similarity and machine learning into weighted Slope One algorithm [D]. Qinhuangdao: Yanshan University, 2015.

[10] Luo Heng. Restricted Boltzmann machines: a collaborative filtering perspective [D]. Shanghai: Shanghai Jiao Tong University, 2011.

[11] Georgiev K, Nakov P. A non-IID framework for collaborative filtering with restricted Boltzmann machines [C]//Proc of International Conference on Machine Learning. 2013: 1148-1156.

[12] He Jieyue, Ma Bei. Based on real-valued conditional restricted Boltzmann machine and social network for collaborative filtering [J]. Chinese Journal of Computers, 2016, 39(1): 183-195.

[13] Huo Shuhua. Research on collaborative filtering based rating prediction algorithm [D]. Beijing: Beijing University of Technology, 2015.

[14] Hinton G E. A practical guide to training restricted Boltzmann machines [R]. Montreal: Department of Computer Science, University of Toronto, 2010.

[15] Kingma D, Ba J. Adam: a method for stochastic optimization [J]. Computer Science, 2014.

[16] Wang Bin. Collaborative filtering recommendation algorithm for converting project attribute similarity and score similarity [D]. Qinhuangdao: Yanshan University, 2017.

[17] Liu Xiaomei, Ouyang Yuanxin, Rong Wenge, et al. Item category aware conditional restricted Boltzmann machine based recommendation [M]//Neural Information Processing. 2015: 609-616.

[18] Du Yongping, Yao Changqing, Huo Shuhua, et al. A new item-based deep network structure using a restricted Boltzmann machine for collaborative fil-

tering [J]. Frontiers of Information Technology & Electronic Engineering, 2017, 18(5): 638-647.

[19] Betru B T, Onana C A, Batchakui B. Deep learning methods on recommender system: a survey of state-of-the-art [J]. International Journal of Computer Applications, 2017, 162(10): 17-22.

Note: Figure translations are in progress. See original paper for figures.