SDAE-LFM: A Latent Factor Model for Recommendation Based on Stack Denoising AutoEncoder

Jianyan Luo, Xing Xing*, Hang Zheng, Mindong Xin and Zhichun Jia

College of Information Science and Technology, Bohai University, Jinzhou, Liaoning 121013, China.
Email: xingxing@bhu.edu.cn

Abstract. Recommendation methods usually associated with data sparsity. The traditional recommendation methods take the users’ rating information as the recommendation basis, which ignore the latent features that can be taking into consideration to model for better recommendations. In order to deal with these problems, we proposed a latent factor model recommendation algorithm based on stack denoising autoencoder (SDAE-LFM), applying Deep Learning technology for latent feature representation learning. A stack denoising autoencoder is applied to extracting feature about item from the label information. Then we factorize the item feature information to perform matrix decomposition training. Finally, we predict the result by the user-item preference matrix. Experimental results on these datasets demonstrate that the proposed recommendation method has better performance.

1. Introduction

With the popularity and evolution of the services and applications of Internet, the extensive information on network has increased so exponentially that information overload [1]. Recommendation algorithms occupy a significant position in solving the information overload [2]. The recommender algorithm analyses the user’s browsing records and user-items [3], mine some potential needs of users, then recommend some contents which the users interested. The traditional recommender algorithms contain mainly content-based recommender algorithm, collaborative filtering recommender algorithm and hybrid recommender algorithm [4-6]. The content-based algorithm finds items which user interested, by using the user-selected and rated items [7]. The collaborative filtering algorithm uses the reaction information about user and items to realize recommendation [8]. The hybrid algorithm integrates both content-based algorithm and collaborative filtering algorithm [9-11].

In present, as a research boom of artificial intelligence, deep learning has made rapid progress in many fields, such as image processing, natural language processing, and speech recognition [6, 12]. Due to its efficient feature extraction and non-linear learning methods, an increasing of researchers is beginning to realize the recommendation algorithm by deep learning [13, 14]. Salakhutdinov et al. used the undirected graph model of the restricted Boltzmann machine to model the user's score and then recommended it, but this method didn’t use the user's potential information [15]. Strub et al. used a stack denoising autoencoder (SDAE) to learn the hidden factors of users and items, and then predicted the missing scores by the hidden factor model [16]. Kim et al. used a convolutional neural network to extract the potential features of related auxiliary information, and combined with the probability matrix for recommendation, but many of the over-fitting in this algorithm need to be manually adjusted [17].

In this paper, we proposed a deep learning algorithm based on stack denoising autoencoder (SDAE-LFM). We apply the stack denoising autoencoder model for learning the item features. The
sparse item matrix is transformed into an item matrix with deep feature information, and then combine with the collaborative filtering algorithm to resolve data sparsity of the traditional collaborative filtering algorithm and achieve better recommendation effect.

2. Related Work

2.1. The Denoising Autoencoder

The autoencoder was studied by Rumelhart et al. in 1988 and gives the definition of an autoencoder [18]. The traditional autoencoder may simply copy the original input, or simply select the reconstruction error, and cannot contain valid feature information [19]. The denoising autoencoder solves this problem by reconstructing input data containing noise [20].

The denoising autoencoder is proposed by Vincent et al [21]. The denoising autoencoder mainly destroys the data by adding a certain degree of noise to the input vector, and then encoding and decoding the input data added with noise. Thus, the learned implicit characteristic variables are more robust [22].

The denoising autoencoder randomly adds noise data to the original input data \( x \) through \( x \sim q_{\theta}(x) \), there by acquiring a partially corrupted data \( x' \), where \( D \) represents the dataset. The denoising autoencoder contains encoder, decoder and hidden layer. The coding layer is the mapping of the noisy version \( x' \) of input \( x \in [0,1]^d \) to the implicit representation \( y \in [0,1]^d \). As the formula 1 shows:

\[
y = f_{\theta}(x) = s(Wx + b)
\]

The set of parameters for this mapping is \( \theta = [W,b] \), \( s \) is a nonlinear function. \( W \) presents a \( d \times d' \) weight matrix, and \( b \) presents the bias unit.

The decoder function \( g_{\theta}(y) \) maps the implicit representation return the rebuilt signification \( z \in [0,1]^d \), as follows:

\[
   z = g_{\theta}(y) = s(W'y + b')
\]

The parameter set \( \theta' = [W',b'] \) of this mapping, the weight matrix \( W' \) of the inverse mapping can be constrained to the transpose of the weight matrix \( W \) : like \( W' = W^T \), which is called the denoising autoencoder weight. \( b' \) is the bias unit.

Each \( x^{(i)} \) is mapped into the \( y^{(i)} \) and a reconstructed signification \( z^{(i)} \), and then the minimum mean reconstruction error is obtained by continuously optimizing the model parameters:

\[
   L(x,z) = \|x - z\|^2
\]

As much as possible, let \( z \) approach the initial input \( x \), and \( z \) presents a function of \( x' \). Our minimized objective function is:

\[
   \min[L(x, g_{\theta}(f_{\theta}(X')))]
\]

2.2. The Stack Denoising Autoencoder

The denoising autoencoders can form a stack denoising autoencoder, and the noise cancellation capability of each layer network is trained by superimposed noise input, so that each layer of the trained encoder can be used for extracting feature with fault tolerance performance while learning [23]. The resulting feature indicates better robustness.

First, the noise data is randomly added to the original data. Then, the reconstruction error is minimized, and the first layer denoising autoencoder is trained to learn the encoder function \( f_{\theta} \). Finally, this function is used to learn the original data, and the obtained result is regarded as the input of the second layer denoising autoencoder to train the second-level denoising autoencoder to learn the...
second-layer encoder function $f^{(2)}_\theta$. This procedure is then iterated until all the denoising autoencoder layers have been trained.

2.3. Latent Factor Model
Latent factor model (LFM) is a widely used algorithm in the recommendation algorithm [24]. Latent factor model first classifies the items, then recommends the classified items according to the user's interest classification. The implicit semantic model predicts the interest of user $u$ about item $i$ as follows:

$$R(u, i) = r_u = p_u^T q_i = \sum_{k=1}^{F} p_{u,k} q_{i,k}$$

(5)

And $p, q$ respectively represent the relationship between the $k$ implicit class and the user interest, and the relationship between the item $i$ and the $k$ implicit class, $F$ is the number of hidden classes, and $r$ is the interest of the user about the item. The loss function is as follows:

$$loss = \sum_{(u,i) \in \mathcal{R}} \left( r_{u,i} - \sum_{k=1}^{F} p_{u,k} q_{i,k} \right)^2 + \lambda \left\| P_u \right\|^2 + \lambda \left\| Q_i \right\|^2$$

(6)

However, LFM algorithm is easy to over-fitting when faced with sparse data, and does not respond well to accurate project feature information.

3. SDAE-LFM Algorithm

3.1. SDAE-LFM Algorithm Implementation
However, LFM algorithm is easy to over-fitting when faced with sparse data, and does not respond well to accurate project feature information. The SDAE-LFM algorithm first extracts the project features from the tag data, and then replaces the implicit factor feature matrix in the LFM algorithm with the feature matrix of the SDAE output. Compared with the traditional LFM algorithm, we convert the project feature matrix and enrich it.

3.2. SDAE-LFM Algorithm Feature Extraction
Since the traditional SDAE network has no way to achieve the score prediction, the data characteristics extracted by SDAE cannot directly reflect the user's preference information for the project. In order to effectively improve the feature quality, we add the sigmoid classifier after the last hidden layer to predict the project. Grading, and the project score as a feedback of effectiveness, turning the parameters, and the optimization function is as follows:

$$\xi = \frac{1}{n} \sum_{j=1}^{n} \left\| x_i - \tilde{x}_i \right\|^2 + \lambda \cdot \frac{1}{n} \sum_{j=1}^{n} \left\| y_i - \tilde{y}_i \right\|^2$$

(7)

$x_i$ is the raw data, $\tilde{x}_i$ is the reconstructed feature of $x_i$, $y_i$ is the item score, $\tilde{y}_i$ is the predicted score produced by the sigmoid, and $\lambda$ is the correction factor.

Construct a label item matrix $F$ in the SDAE-LFM algorithm. $R_{ij}$ represents the rating of user $i$ and item $j$, $F_{ij}$ represents the value of the item $j$ to the label $i$, and the joint matrix $F$ and $R$, and then obtain the target matrix $P$.

$$P_{ij} = \frac{1}{N} \sum_{t} R_{ij} F_{ij}$$

(8)
After constructing the target matrix $P$, the collaborative filtering algorithm is improved by using the original scoring matrix $R$ and the newly constructed target matrix $P$, and the target matrix $Q$ is integrated into the collaborative filtering algorithm to construct a new loss function.

$$L_{t} = \frac{1}{2} \sum_{i} \sum_{j} (R_{ij} - U_{i}^{T}V_{j})^2 + \frac{\alpha_{U}}{2} \sum_{i} \|U_{i}\|^2 + \frac{\alpha_{V}}{2} \sum_{j} \|V_{j}\|^2 + \frac{\alpha_{Q}}{2} \sum_{i} \|Q_{i}\|^2 + \frac{\alpha_{L}}{2} \sum_{i} \sum_{j} (F_{ij} - U_{i}^{T}Q_{j})^2$$

(9)

4. Experiment and Result

4.1. Metrics
In this paper, we evaluate the proposed algorithm by using RMSE and Recall @ K. RMSE evaluates the proximity of the user’s predicted score to the actual score, as a basis for evaluating the performance. Recall @ K measures the recall rate of the algorithm, which is the ratio of the number of recommended results and the total number of correct results.

4.2. Datasets
This paper verifies the proposed algorithm on MovieLens 1M and MovieLens 10M. The datasets are divided into 10 groups, each time selecting 2 groups as test-sets, and the remaining 8 groups as train-sets. 10 tests to ensure that each data participates and only participates in one test. The detailed data information is displayed in the following table.

| Dataset     | Movielens 1M | Movielens 10M |
|-------------|--------------|---------------|
| User        | 6,040        | 71,567        |
| Item        | 3,900        | 10,681        |
| Rating      | 1,000,209    | 10,000,054    |
| Sparsity    | 95.8%        | 98.6%         |
| User Feature| Age, Gender, Occupation | Age, Gender, Occupation |
| Item Feature| Title, Genres | Title, Genres |

4.3. Comparative Analysis of Experimental Results
This paper is related to the traditional collaborative filtering recommendation algorithm (PMF) [25], the deep learning algorithm (DBN) without tags [26], and the recommendation algorithm (PACE) combined with tag information [27].
The results shown in Figure 1 are the Recall @ K values of the four different algorithms. Obviously, the PMF which does not attract auxiliary information is worst. Both DBN and PACE use some auxiliary information, and take an improved performance. And the SDAE-LFM algorithm outperforms than the other three algorithms.

Figure 2 illustrates the different RMSE performance of the four models. From the data contrast, the SDAE-LFM we proposed takes the best efficiency on the sparsely powerful MovieLens datasets.

5. Conclusions
Aiming at the low recommend accurateness of the traditional recommendation algorithm on the sparse data set, we proposed an improved LFM algorithm which based on stack noise reduction self-encoder, which uses the self-encoder to extract data features and implicit factor matrix decomposition algorithm. By comparing with PMF, DBN and PACE, the training time of SDAE-LFM algorithm has not increased significantly while improving the recommendation accuracy. The results evidence that the SDAE-LFM algorithm could process large-scale data in a reasonable time and effectively improve recommended performance.

6. Acknowledgments
This paper is partially supported by the National Natural Science Foundation of China under Grant No.61972053, by the Scientific Research Foundation of Liaoning Education Department under Grant
No. LQ2019016, No. LJ2019015, and by the Natural Science Foundation of Liaoning Province, China under Grant No.2019-ZD-0505, and by the Graduate Education and Teaching Reform Project of Bohai University under Grant No. YKC201901.

7. References

[1] Resnick P and Varian H R 1997 Recommender systems Communications of The ACM 40 56-58.
[2] Herlocker J L Konstan J A Terveen L and Riedl J 2004 Evaluating collaborative filtering recommender systems ACM Transactions on Information Systems 22 5-53.
[3] Verbert K Manouselis N Ochoa X Wolpers M Drachsler H Bosnic I and Duval E 2012 Context-Aware Recommender Systems for Learning: A Survey and Future Challenges IEEE Transactions on Learning Technologies 5 318-335.
[4] Campos P G Diez F and Cantador I 2014 Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols User Modeling and User-adapted Interaction, 24 67-119.
[5] Adomavicius G Mobasher B Ricci F and Tuzhilin A 2011 Context-Aware Recommender Systems Ai Magazine 32 67-80.
[6] Wei J He J Chen K Zhou Y and Tang Z 2017 Collaborative filtering and deep learning based recommendation system for cold start items Expert Systems With Applications 69 29-39.
[7] Park D H Kim H K Choi I Y and Kim J K 2012 A literature review and classification of recommender systems research Expert Systems With Applications 39 10059-72
[8] Errico J H Sezan I M Borden G R Feather G A and Grover M G 2015 Collaborative recommendation system US.
[9] Zhang F Yuan N J Lian D Xie X. and Ma W 2016 Collaborative Knowledge Base Embedding for Recommender Systems knowledge discovery and data mining 353-62.
[10] Yanhong G and Guishi D 2008 Hybrid Recommendation Algorithm of Item Cold-start in Collaborative Filtering System Computer Engineering.
[11] Burke R 2002 Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-adapted Interaction 12 331-70.
[12] Zhang S Yao L Sun A and Tay Y 2019 Deep Learning Based Recommender System: A Survey and New Perspectives ACM Computing Surveys 52.
[13] Lecun Y Bengio Y and Hinton G E 2015 Deep learning Nature 521 436-44.
[14] Zheng L Norouzi V and Yu P S 2017 Joint Deep Modeling of Users and Items Using Reviews for Recommendation web search and data mining 425-34.
[15] Salakhutdinov R Mnih A and Hinton G E 2007 Restricted Boltzmann machines for collaborative filtering international conference on machine learning 791-98.
[16] Strub F and Mary J 2015 Collaborative Filtering with Stacked Denoising AutoEncoders and Sparse Inputs neural information processing systems.
[17] Kim D Park C Oh J Lee S and Yu H 2016 Convolutional Matrix Factorization for Document Context-Aware Recommendation conference on recommender systems 233-40.
[18] Rumelhart D E Hinton G E and Williams R J 1988 Learning representations by back-propagating errors Nature 323 696-99.
[19] Strub F Gaudel R and Mary J 2016 Hybrid Recommender System based on Autoencoders conference on recommender systems 11-16.
[20] Wu Y Dubois C Zheng A X and Ester M 2016 Collaborative Denoising Auto-Encoders for Top-N Recommender Systems web search and data mining 153-62.
[21] Vincent P Larochelle H Bengio Y and Manzagol P 2008 Extracting and composing robust features with denoising autoencoders international conference on machine learning 1096-103.
[22] Masci J Meier U Ciresan D and Schmidhuber J 2011 Stacked convolutional auto-encoders for hierarchical feature extraction international conference on artificial neural networks 52-59
[23] Vincent P Larochelle H Lajoie I Bengio Y and Manzagol P 2010 Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion Journal of Machine Learning Research 3371-408.
[24] Jenatton R Roux N L Bordes A and Obozinski G 2012 A latent factor model for highly multi-relnational data *neural information processing systems* 3167-75  
[25] Mnih A and Salakhutdinov R 2007 Probabilistic Matrix Factorization *neural information processing systems* 1257-64  
[26] Chen Y Zhao X and Jia X 2015 Spectral–Spatial Classification of Hyperspectral Data Based on Deep Belief Network *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 8 2381-92.  
[27] Yang C Bai L Zhang C Yuan Q and Han J 2017 Bridging Collaborative Filtering and Semi-Supervised Learning: A Neural Approach for POI Recommendation *knowledge discovery and data mining* 1245-54.