Online News Recommender Based on Stacked Auto-Encoder

Sanxing Cao, Nan Yang
New Media Institute
Communication University of China
Beijing, China
sxcao@cuc.edu.cn, yangnan@cuc.edu.cn

Zhengzheng Liu
Information Engineering Institute
Communication University of China
Beijing, China
838789694@qq.com

Abstract—**Because of the popularity of Internet and mobile Internet, people are facing serious information overloading problems nowadays. Recommendation engine is very useful to help people to reach the Internet news they want through the network. Collaborative filtering (CF), such as item-based CF, is the most popular branch in recommendation domain. But the data's high-dimension as well as data sparsity are always the main problems. A novel CF method is introduced in this article, which uses stacked auto-encoder with denoising, an unsupervised deep learning method, to extract the useful low-dimension features from the original sparse user-item matrices. Together with proper similarity computing algorithms, the method provided in this article is proved to be more precise than the methods based on SVD or item-based CF..**

Keywords—recommendation engine; collaborative filtering; deep learning; stacked auto-encoder

I. Introduction

Because of the development of Internet and mobile Internet, more and more people use mobile devices to access the network. According to the data from CNNIC [1], in China alone there are almost 700 million people using mobile devices to access the internet. Every day many people use mobile to read news in their fast paced daily life. How to help the reader to reach the content as soon as possible is a very important issue not only to readers but also to news media.

Because New Media Institute is a main stream media research institute in domestic media industry, many content providers want it to provide a specified platform, for them to get the comments about their contents within the industry, so they can improve. In the platform, the data fetching part utilizes mainstream searching technology plus distributed implementation, and the data analyzing part implements robust machine learning technology to fulfill the data mining [2]. Research on recommender algorithm is also conducted. Recommendation mechanism relies on the concept of the long tail phenomenon and theory. According to the long tail theory, items that are usually neglected or less interested could earn equal or more values compared with popular ones, when their presentation and production chances are high and costs are low. This means less popular items or goods could achieve good attention and sales in Internet where information of almost all the commodity could be easily presented to users or customers.

The development team tries to use the latest techniques to improve the algorithms, and improve the platform's service ability towards the users step by step. Deep learning is very helpful in its ability to learn the features and to do comparison. Stacked denoising auto-encoder, which have powerful characteristics of unsupervised learning and feature extraction ability, has been experimented to improve the platform's recommendation efficiency.

There are mainly five parts in this article: 1) Introduction, 2) Main Methods, 3) Results, 4) Discussion, 5) Conclusion.

II. Main Methods

A. Recommender overview

Recommender is an active research field in data mining and machine learning for some time. There exist different approaches for recommendation engines, such as content-based filtering, collaborative filtering, and hybrid techniques [3].

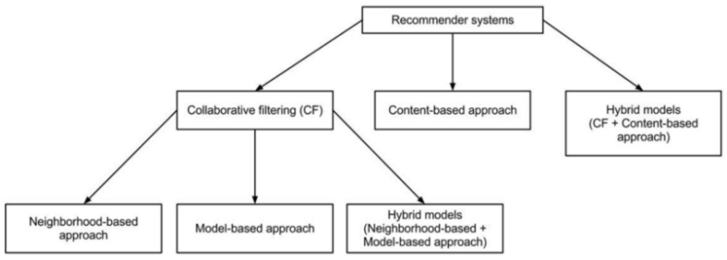
Recommender systems
Collaborative filtering (CF)
Content-based approach
Hybrid models (CF + Content-based approach)
Neighborhood-based approach
Model-based approach
Hybrid models (Neighborhood-based + Model-based approach)

Fig. 1. Recommendation system and collaborative filtering

B. Collaborative filtering

Collaborative filtering (CF) is one of the most efficient ones. There are also different kinds of CF approaches, one way is to distinguish them as neighborhood-based algorithms model-based algorithms and hybrid models [4, 5]. CF methods are described like this: they are based on collecting and analyzing a large amount of information on users' behaviors, activities or preferences and predicting what users would like based on their similarity to other users. Figure 1 above from Wikipedia demonstrates the structure of recommendation systems and collaborative filtering [6]. In CF, the information on personal preferences, tastes, and quality are all carried in (explicit or implicit) user ratings which can be utilized [7].

Neighborhood-based CF techniques, which can also be called memory-based approach, means using the entire or a fraction of user's operation record, usually the user-item database in the system to generate the a prediction or preferences for the users [5, 8]. Item-based CF currently is the most popular approaches. Neighborhood-based CF techniques can also be classified specifically as user-based CF filtering, and item-based CF filtering [11]. They can be used to produce a prediction for the active user by taking the weighted average of all the ratings of the user or item on a certain item, or using a simple weighted average [4]. Item-based recommendation method is more popular currently [12].

Model-based CF are developed to make predictions using data mining, machine learning algorithms to find patterns based on training data. There are many model-based CF methods, including Bayesian networks, clustering models, latent semantic models, and newly RBM, Auto-Encoder based solutions.

Hybrid methods combine the memory-based CF and the model-based CF algorithms, trying to make recommendation more effectively. However, sometimes these approaches have increased complexity and are expensive to implement [9, 10].

In the algorithm aspect, this article focused on model-based CF recommendation systems, and compare it with item-based and SVD based methods [14].

C. Stacked auto-encoder for recommender

Model-based approaches are those algorithms, which do the computing with predefined offline model, but presenting the result online. The design and development of models (such as machine learning, deep learning algorithms) allow the system to learn to recognize complex patterns based on the training data, and then make intelligent predictions for the collaborative filtering tasks for test data or real-world data, based on the learned models.

There are several main stream model-based algorithms in use, such as Bayesian Net or Dependency Net algorithms [15-17], Clustering algorithms [18], Markov decision processes [19], Regression based algorithms [20-22], SVD based algorithms, and Auto-Encoder based algorithms [23]. Model-based algorithms have been investigated to solve the shortcomings of memory-based CF algorithms [11, 14].

Since deep learning has been a hot spot in machine learning, and improved the performance in pattern recognition and NLP greatly, it has been used in recommendation domain certainly [24, 25]. Stacked auto-encoder based CF is one of the deep learning approaches.

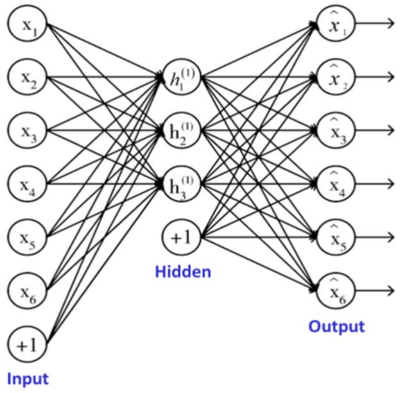

Fig. 2. The general structure of an Auto-Encoder

Now people are trying to use deep learning to improve the efficiency and performance of recommendation systems. Because of the powerful feature extraction ability of Stacked denoising Auto-Encoder (SDAE), it is the well-suited method in recommendation engine.

Auto-encoder, which was mentioned by Rumelhart in 1986 [26, 27], is a kind of neural network and unsupervised learning methods which attempts to be trained to copy its input to output. It is typically a three layers neural network. The first layer is the input layer, the second layer is the hidden layer, and the third layer is the output layer, referring to Fig. 2. The mechanisms between the input layer and hidden layer could learn a dataset's low dimensional distributed representations from input layer's original high dimensional counterparts, just like an encoder. The mechanisms between the hidden layer and output layer just do the opposite things, which can reconstitute the original high dimensional information, just like a decoder.

When using auto-encoder alone, the features extracted by the encoder are not robust enough. After adding the Gaussian noise, it is better [28]. So for making the hidden layer to discover more robust features and to prevent it from simply learning the identity, denoising auto-encoder were created. It is a stochastic kind of the auto-encoder which does two things, trying to encode the input (preserve the information about the input), and trying to undo the effect of a corruption process stochastically applied to the input of the auto-encoder. Followed structure is a denoising auto-encoder [29]:

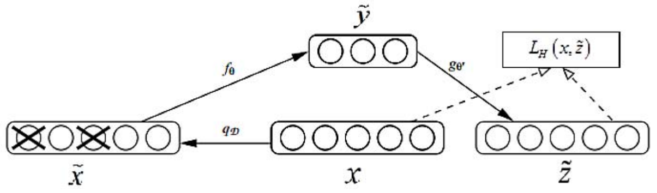

Fig. 3. The principle of denoising Auto-Encoder

A stacked auto-encoder is a neural network consisting of multiple layers of sparse auto-encoders in which the outputs of each layer is wired to the inputs of the successive layer [30]. Denoising auto-encoders can be stacked to form a deep network by feeding the latent representation (output code) found on the layer below as input to the current layer [31].

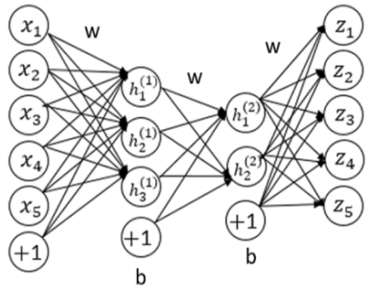

Fig. 4. The structure of a Stacked denoising Auto-Encoder

D. Algorithm designing

Online videos and games show a long-tailed pattern, so they are suitable to be handled in recommendation. Here model-based CF and stacked auto-encoder was chosen as the main methods in this recommendation system design.

The main procedure is like this: a) using stacked auto-encoder to extract the low dimensional features from the sparse high dimensional user-movie relationship; b) calculating the similarity with cosine measurement; c) achieving the top-N recommendation [29].

1) Pretreatment of data

The system uses gradient decent to train the neural networks. In order to quicken the train process, data pretreatment is necessary. The training dataset is Movielens 1M. If marked the user set as U, movie set as M. If taking the remarks from user U towards movie M as r_m^u, it is an integer ranging from 1-5. Taking r_{min} as the lowest remarks, and r_{max} as the highest remarks, the pretreatment can be achieved by:

$$\boldsymbol{r_m^u} = \frac{r_m^u - r_{min}}{r_{max} - r_{min}} \tag{1}$$

2) Training of SDAE structure

By above Figures, for every layer L, after the input x adding the noise as $\tilde{x}$, the output of hidden layer is as followed:

$$\boldsymbol{h^L = \sigma(w^L\tilde{x} + b)} \tag{2}$$

The training of the network is layer-wised, which means Stacked auto-encoder uses layer-wise greedy method to train the networks, layer by layer [29]. Each decoder for training is as followed:

$$\boldsymbol{\hat{x} = \sigma(w^L h^L + b'^L)} \tag{3}$$

In above two functions, σ stands for sigmoid activation function:

$$\boldsymbol{\sigma(t) = \frac{1}{1-e^{-t}}} \tag{4}$$

If the input is interpreted as either bit vectors or vectors of bit probabilities, then the reconstitution should use cross-entropy cost function. The cost function for single sample is as followed:

$$\boldsymbol{L_H(x,z) = -\sum_{k=1}^{d}[x_k \log \hat{x}_k + (1 - x_k)\log(1 - \hat{x}_k)]} \tag{5}$$

For a training set which has S samples, the total cost function is as followed:

$$\boldsymbol{J(\theta) = -\frac{1}{S}[\sum_{i=1}^{S}\sum_{k=1}^{U} x_k^{(i)} \log(\hat{x}_k^{(i)}) + (1 - x_k^{(i)})\log(1 - \hat{x}_k^{(i)})] + \frac{\lambda}{2U}\sum_{l=1}^{L-1}\sum_{i=1}^{S_l}\sum_{k=1}^{S_{l+1}}(\theta_{ki}^{(l)})^2} \tag{6}$$

Stochastic Gradient Descent (SGD) and *Back Propagation,* have been used for the training of the auto-encoders. For the training of hidden layers, it is layer wised:

$$\boldsymbol{\nabla W^l = \delta^l (x^i)^T, \nabla b^l = \delta^l} \tag{7}$$

$$\boldsymbol{\nabla W^i = \delta^i (h^i)^T, \nabla b^i = \delta^i} \tag{8}$$

The encoder's effectiveness is similar with PCA (Principal Component Analysis). But if the hidden layer is non-linear, the auto-encoder behaves differently from PCA or even better.

3) Similarity computing

SDAE in this system is powerful for extracting the features of the users and items. After extraction of the features, similarity calculating between platform users will be based on these results. Here similarity was defined as a utility matrix. In the sense of collaborative filtering, users are considered similar if their behavior are similar, that is to say, user vectors in utility matrix are close according to the distance measure.
In similarity calculating, also three specific steps can be adopted:

- Calculating and assign weights to all the users with respect to similarity towards the active user.
- Neighborhood calculation, choose n users who have the highest similarity with the active user.
- From the weighted integration of the chosen neighbors' ratings, calculate the needed prediction [32].

Here in a user-based CF algorithm, the calculating of the similarity, which is described as $Simil_{a,u}$ between the users α and u who have both rated the same items, is one of the essential steps. There are many methods to calculate the similarity, and methods such as Pearson correlation, Cosine are the most commonly used [33]. In this case, Cosine is used. For the specific algorithms to calculate the similarity, the cosine method could be more convenient. One can treat the ratings of two users as a vector in an m-dimensional space, and compute similarity based on the cosine of the angle between them, which is given by:

$$\boldsymbol{Simil_{a,u} = COS(\overrightarrow{r_a}, \overrightarrow{r_u}) = \frac{\sum_{i=1}^{m} r_{a,i} \cdot r_{u,i}}{\sqrt{\sum_{i=1}^{m} r_{a,i}^2}\sqrt{\sum_{i=1}^{m} r_{u,i}^2}}} \tag{9}$$

Predictions are generally computed as the weighted average of deviations from the neighbor's mean, as given by [26]:

$$\boldsymbol{Pred_{a,i} = \bar{r}_a + \frac{\sum_{u \in J}(r_{u,i} - \bar{r}_u) \times Simil_{a,u}}{\sum_{u \in J} Simil_{a,u}}} \tag{10}$$

Where $Pred_{a,i}$ is the prediction for the active user α for item i, $simil_{a,u}$ is the similarity between users α and u, and J is the neighborhood or set of most similar users.

In order to favor users with the high similarity to the active user [9, 26], case amplification was introduced to transform the original weights in function (3) to:

$$\boldsymbol{Simil'_{a,u} = Simil_{a,u} \cdot |Simil_{a,u}|^{\rho-1}} \tag{11}$$

In which ρ is the amplification factor, and ρ ≥ 1.

4) Top-N recommendations

Top-N recommendation is to recommend a set of N top-ranked items that would be of interest to a certain user. The result of the last similarity calculation can be used to make top-N recommendations [11, 34].

Generally the number of users in the computation of similarity is regarded as the neighborhood size of the active user [4]. When the task is to generate a top-N recommendation, we need to find k most similar users or items (nearest neighbors) after computing the similarities, then to integrate the neighbors to get the top-N most frequent items as the recommendation.

Here defined R as an n × m user-video matrix containing historical viewing or commenting information of n users on m videos. In this matrix, $r_{i,j}$ should be set to one if user-i has played the video-j, otherwise it should be set to zero. Let U be the set of videos that have already been viewed by the user for which the system want to compute and give the top-N recommendations. Here the user could be considered as the active user, and in order to simplify the presentation, the active user can be assumed not belonging to the n users stored in matrix R. Then recommendation algorithms compute the top-N recommended videos for that user [35-37].

The procedure for generating top-N recommendation is as followed (simplified):

- Taking the user-remarked movies as a dataset, named Mu; calculating the similarity between the movies in Mu; building the nearest collections with length K.
- Combining the K nearest dataset as M*, except the user remarked movies.
- Calculating the similarities between Mu and M* one by one.
- Choosing the top-N similarity movies to recommend to users.

III. Result

A. *Dataset for training*

In real situation, the recommendation engine uses the data collection from the platform. But for experimenting, some standard datasets such as MovieLens, Netflix and Yelp, are chosen to develop the algorithm and measure the precision. Here MovieLens 1M dataset is chosen for the offline training and testing.

MovieLens 1M dataset includes abundant information, such as the scoring data of about 6000 users towards about 4000 movies. Each rating in MovieLens is an integer between 1 to 5, which means the worst and the best. These ratings are highly sparse. But in this case, we didn't consider the side information. It was divided as 80% as training data and 20% as testing data.

B. *Recommendation evaluation*

There are several ways to determine the accuracy of a recommendation system, such as the evaluation of Predictive accuracy, the evaluation of Top-N recommendation, etc.

In the evaluation of Predictive accuracy, the quality of a recommendation system can be evaluated by comparing recommendation results to a test set of known user ratings. The most commonly used method in the test is Mean Absolute Error (MAE) [32]. However, in the evaluation of top-N recommendation, precision and recall are defined for the measurement of the recommendation efficiency [38]. Here, Take Te(u) as the collection which has been remarked by users in test set, L(u) as the action of users:

$$\textbf{Precision} = \frac{\sum_{u\in U} |L(u) \cap Te(u)|}{\sum_{u\in U} |L(u)|} \tag{12}$$

$$\textbf{Recall} = \frac{\sum_{u\in U} |L(u) \cap Te(u)|}{\sum_{u\in U} |Te(u)|} \tag{13}$$

Because top-N recommendation is adopted in this system, precision and recall are chosen as the measurement. Compared with SVD method and item-based CF, the stacked denoising auto-encoder method has achieved better result.

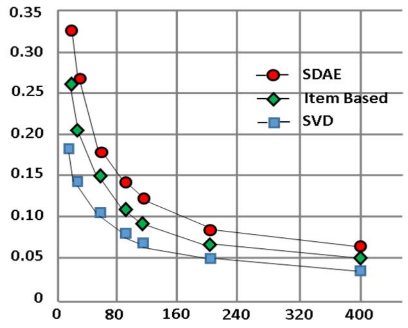

Fig. 5. Comparison of Precision

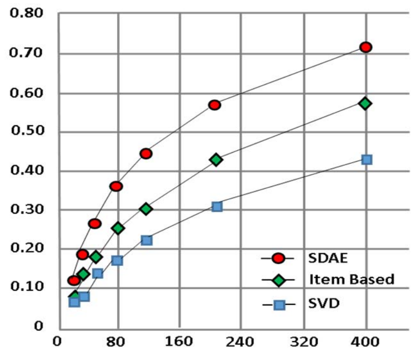

Fig. 6. Comparison of Recall

IV. Discussion

Auto-encoder acts like they copy the input to the output. But the interesting part is the hidden layer, in which the dimension is usually lower than the input layer. If the output layer can rebuild the input, and the hidden layer worked in nonlinear way, that means the hidden layer has get a new representation of the input, similar with PCA but better.

Being stacked, deep learning showed its power, that auto-encoder has even more power in feature extraction ability. After adding the Gaussian noise, the feature extracted is very robust. Just like the expectation, the method provided in this article is more precise than SVD or item-based CF methods.

The success of machine learning algorithms generally depends on data representation. Different representations can entangle and hide some of the different explanatory factors of variation behind the data. Learning representations of the data can make it easier to extract useful information when building classifiers or other predictors [39]. Among the various ways of learning re-presentations, such as Translation-based model or other related methods, should be paid much attention [40].

It treats the recommendation problem as a representation learning one, by computing representations for users and items in a common latent space, and it could give new approaches to handle problems such as sparsity or cold-start in recommender [41]. Given these representations, traditional approaches can compute missing ratings made by a user u over an item i as the dot product, such that the more similar the user and item representations are, the higher the predication rate can be [42]. Representation learning will be the future choice to improve our recommendation system.

V. Conclusion

The algorithm works well. A system related was built since 2014 and improved little by little, and could operate properly. It also has some relationships with research on online short videos comments analyzing and utilization within the institution, which give the analysis result data from the user's comments toward short videos [2].

Acknowledgment

This paper is partly supported by "Research on Technology of Big Data" project, which is funded by the New Media Institute of Communication University of China (14NM06). Thanks for all the referees, for their valuable review and help to the paper.

References

[1] "The 38th CNNIC Report, http:// www.cnnic.net.cn /hlwfzyj/ hlwxzbg/ hlwtjbg / 201608/t20160803_54392.htm.

[2] N. Yang, S. Cao,S. Zhang, Data Analysis System for Online Short Video Comments, in Proceedings of the 15th IEEE/ACIS International Conference on Computer and Information Science, 2016.

[3] J. Sobecki, K. Piwowar, Comparison of Different Recommendation Methods for an e-Commerce Application, Asian Conference on Intelligent Information and Database Systems (IEEE ACIIDS), 2009.

[4] X. Su, T. Khoshgoftaar, A survey of collaborative filtering techniques, Advances in Artificial Intelligence archive, 2009.

[5] J. Schafer1, D. Frankowski, J. Herlocker, S. Sen, Collaborative Filtering Recommender Systems, International Conference on Intelligent Systems Design& Applications, 2015, pp. 438-443.

[6] Collaborative filtering, https: // en.wikipedia.org / wiki / Collaborative _filtering.

[7] K. Yu, A. Schwaighofer, V. Tresp, X. Xu, H. Kriegel, Probabilistic Memory-based Collaborative Filtering, IEEE Transactions on Knowledge & Data Engineering, 2004, pp. 16(1):56-69.

[8] D. Goldberg, Using collaborative fi1tering to weave an information tapestry, Communications of the Acm, 1992, pp. 35(12):61-70.

[9] M. Pazzani, A framework for collaborative, content-based and demographic filtering, Artificial Intelligence Review, 1999, pp. 13:393–408.

[10] R. Burke, Hybrid Web Recommender Systems, Adaptive Web: Methods & Strategies of Web Personalization, Lecture notes on computer science, 2007, pp.4321:377-408.

[11] J. Breese, D. Heckerman, and C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, 1998.

[12] B. Sarwar, G. Karypis, J. A. Konstan, J. Riedl, Itembased collaborative filtering recommendation algorithms, in Proceedings of the 10th International Conference on World Wide Web (WWW '01), 2001, pp. 285–295.

[13] G. Karypis, Evaluation of item-based top-N recommendation algorithms, in Proceedings of the International Conference on Information and Knowledge Management, 2001, pp. 247–254.

[14] D. Billsus, M. Pazzani, Learning Collaborative Information Filters, Machine Learning In: Fifteenth International Conference, 1998.

[15] C. Basu, H. Hirsh, W. Cohen, Recommendation as classification: using social and content-based information in recommendation, Fifteenth National/tenth Conference on Artificial Intelligence/innovative Applications of Artificial Intelligence, 1998.

[16] S. Andersen, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Journal of Philosophy, 1991, pp. 48(1):117-124.

[17] K. Miyahara,M. Pazzani, Improvement of collaborative filtering with the simple Bayesian classifier, Ipsj Journal, 2002, pp. 43:3429-3437.

[18] J. Han, M. Kamber, J. Pei, Data Mining: Concepts and Techniques: Third Edition, ELSEVIER, 2012.

[19] G. Shani, D. Heckerman, R. Brafman, An MDP-Based Re-commender System, Journal of Machine Learning Research, 2015, pp. 1265-1295.

[20] J. Canny, Collaborative filtering with privacy via factor analysis, International Acm Sigir Conference on Research & Development in Information Retrieval, 2002.

[21] S. Vucetic,Z. Obradovic, Collaborative Filtering Using a Regression-Based Approach, Knowledge & Information Systems, 2005.

[22] Y. Koren, R. Bell, C. Volinsky, Matrix Factorization Techniques for Recommender Systems, Computer, 2009, 42(8):30-37.

[23] Y. Wu, C. Dubois, A. Zheng, M. Ester, Collaborative Denoising Auto-Encoders for Top-N Recommender Systems, Acm International Conference on Web Search & Data Mining, 2016.

[24] H. Wang, N. Wang, D. Yeung, Collaborative Deep Learning for Recommender Systems, doi.acm.org, 2015.

[25] F. Zhang, N. Yuan, D. Lian, X. Xie, W. Ma, Collaborative Knowledge Base Embedding for Recommender Systems, Acm Sigkdd International Conference on Knowledge Discovery & Data Mining, 2016.

[26] D. Rumelhart, G. Hinton, R. Williams, Learning representations by back-propagating errors, MIT Press, 1986.

[27] G. Hinton, R. Zemel, Autoencoders, minimum description length, and Helmholtz free energy, NIPS'1993.

[28] P. Vincent, H. Larochelle, Y. Bengio, P. Manzagol, Extracting and composing robust features with denoising autoencoders, ICML '08 Proceedings of the 25th international conference on Machine learning, 2008, pp:1096-1103.

[29] Y. Zhou, J. Chen, Stacked denoising autoencoder for collaborative filtering algorithm, Application Research of Computers, 2016.

[30] http://ufldl.stanford.edu/wiki/index.php/Stacked_Autoencoders.

[31] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P. Manzagol, Stacked Denoising Autoencoders Learning Useful Representations in a Deep Network with a Local Denoising Criterion, Journal of Machine Learning Research, 2010, 11(12), 3371-3408.

[32] P. Melville, V. Sindhwani, Recommender Systems, Encyclopedia of Machine Learning, 2010.

[33] M. McLaughlin, J. Herlocker, A collaborative filtering algorithm and evaluation metric that accurately model the user experience, in Proceedings of 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Sheffield, UK, 2004, pp. 329–336.

[34] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Analysis of recommendation algorithms for E-commerce, in Proceedings of the ACM E-Commerce, Minneapolis, Minn, USA, 2000, pp. 158–167.

[35] Y. Ling,D. Guo,F. Cai,User-based Clustering with Top-N Recommendation on Cold-Start Problem, Third International Conference on Intelligent System Design and Engineering, 2013, pp. 1585-1589.

[36] G. Karypis, Evaluation of item-based top-N recommendation algorithms, in Proceedings of the International Conference on Information and Knowledge Management, Atlanta, Ga, USA, 2001, pp. 247–254.

[37] G. Salton, Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer, AddisonWesley, 1989.

[38] C. Yu, Y. Zhu, C. Jin, F. Lu, Collaborative Filtering Algorithms Based on Auto-encoders, MicroComputer Application (in Chinese), Vol.31, No.11, 2015.

[39] Y. Bengio, A. Vincent, Representation learning: A review and new Perspectives, IEEE Trans on Pattern Analysis and Machine Intelligence, 2013.

[40] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating Embeddings for Modeling Multi-relational Data, Advances in Neural Information Processing Systems 26 (NIPS 2013), 2013.

[41] G. Contardo,L. Denoyer,T. Artieres, Representation Learning for cold-start recommendation, Eprint Arxiv, 2015.

[42] F. Zhang, N. Yuan, D. Lian, X. Xie, W. Ma, Collaborative Knowledge Base Embedding for Recommender Systems, Acm Sigkdd International Conference on Knowledge Discovery & Data Mining, 2016.