An Adaptive Clustering and Incremental Learning Solution to Cold Start Problem in Recommendation Systems

Sriram G. Sanjeevi, Srikanth. Grandhe, Himanshu. Shrivastav, Ankit. Yadav, Tapas Kumar. Mahanta, and Ch. Naveen Kumar

Abstract**—**Recommendation systems are the systems that recommend items to the users based on their tastes and interest. With the expansion of data on the web and a variety of products available in the E-Commerce sector, the recommender systems come to the aid of the users. The cold start problem is one of the major drawbacks for the recommender system and can prove costly for its success if proper steps are not taken. We solve this by proposing variations in adaptive clustering methods that will be used in the pre-processing stage to generate data only for lowly ranked items so that they can compete with the highly ranked items. We also develop a predictor based on incremental learning algorithm based on LEARN++ that meets the needs of the adaptively clustered data.

Keywords**—**Adaptive Clustering, Cold Start, Incremental learning, Recommendation System, Slope One predictor

I. Introduction

RECOMMENDATION systems are the systems that recommend items to the users based on their tastes and interest. With the expansion of data on the web and a variety of products available in the E-Commerce sector, the recommender systems come to the aid of the users. Some of the websites that use recommender systems are Amazon.com and Ebay.com for product sales, Google and Yahoo! for advertisements, IMDB.com and MovieLens.com for movie recommendations and so on.

Recommender systems have become an important research area since the appearance of collaborative filtering [1, 2, 4, 6] in the mid 1990s. The interest in this area still remains high because it constitutes a problem rich research area and because of the plethora of practical applications that help the users to deal with vast information and provide personalized recommendations.

Sriram G. Sanjeevi, is with the Department of Computer Science and Engineering, National Institute of Technology, Warangal, Telangana State, 506004 India (corresponding author's phone: +91-870-2430440 ; fax: +91-870-2459547; e-mail: sgsanjeevi108@gmail.com).

Srikanth. Grandhe , Himanshu. Shrivastav, Ankit. Yadav, Tapas Kumar. Mahanta, and Ch. Naveen Kumar are with Department of Computer Science and Engineering, National Institute of Technology, Warangal, Telangana State, 506004 India.

The standard techniques applied for standard filtering techniques are not appropriate for the cold start problem as they do not take this specific problem into consideration and are designed for general application.

The cold start problem [3, 4, 5, 7] is one of the major drawbacks for the recommender system and can prove costly for its success if proper steps are not taken. When a new user enters the system and does not find recommendations that are not convincing him, the user may not reuse the system and market value of the system drops down. Some alternatives need to be framed out to handle the new customers and keep the performance levels high. Similarly, for a new item introduced in the system, if it does not get make it to the recommendations of users the suppliers will lose trust on the system and will not incorporate it in their businesses. For a new item or items with less data, they will never fall in the top-k recommendations to the user even if they are desired in the recommendations and hence their rating count from the user would never improve.

The cold start problem for items is popularly known as the long tail problem as the items in the item list can be classified into two portions called head and tail. The head bears the items with large number of ratings from the user while the tail bears the remaining items with insufficient ratings. By the study of real life datasets it has been learnt that the number of items in the tail are far greater than the items in the head and hence the name long tail problem is popular.

II. Related Work

The problem of cold start has been previously studied by many researchers and solutions have been provided. Despite the advances in the recommendation area, cold start problem is still unsolved. This is due to the fact that the solutions provided are limited to one particular system and the parameters and performance varies from system to system. It emerges each time a new item or new user enters the system. In particular, Ungar and Foster[10] cluster similar users into the same group to overcome the data sparsity problem for collaborative filtering. In addition, item clustering is also used to improve the prediction the accuracy of collaborative filtering in which items are divided into smaller groups, and existing algorithms are used on each group separately.

Many researchers have tried to tackle this problem in many ways and tried to solve using Data Mining Techniques such as associative rule mining and many different mining algorithms that could be used to cover the data sparsity problem[7, 8].

The cold start problem for the new items also known as the Long Tail problem[3, 5] is more widely studied and solutions with reasonable performance have been provided compared to its counterpart. Few popular solutions use content based methods to improve accuracy and reduce error rates.

In recent work related to this study, Park and Tuzhilin[5] use clustering data ideas to solve the long tail problem as well as cutting the item set into head and tail parts and apply clustering technique called expectation-maximization in the tail part. This method is referred as clustered tail recommendation method. However, the clustered tail method does not adaptively cluster the items and the head or tail splitting points need to be located manually.

III. Related Concepts

A. Adaptive Clustering method

In order to keep the good features and remove the bad features of total clustering[3], a method called Adaptive clustering is introduced in [3].

The Adaptive clustering method clusters the items with similar items when it has only a small amount of data, but groups to a lesser extent or does not group at all when it has sufficient amount of data. This clustering process takes an intermediate step between the each item method and the total clustering method.

Algorithm: Adaptive Clustering

Input: α -> minimum sum of ratings to form a cluster

Cluster set C formed using k-means

Output: Hash table H containing list of items that are adaptively clustered with each item

1.**for** each cluster C_i, **do**

2. **for** each item x_k ε C_i**do**

3. **if** number of ratings of x_k> α**then**

4. do nothing

5. **else do**

6. Ratings_count= x_k.count

7. **while**(ratings_count< α)**do**

8. find movies x_j ε C_i such that

x_j .count < α and x_j is not in the list of x_k

9. add the movie $x_j$to list of x_k in the hash table H that is most similar tox_k.

10. ratings_count = ratings_count + x_j.count

11. **end while**

12. **end for**

13.**end for**

B. Incremental Learning(LEARN++)

Incremental learning is a learning mechanism that allows classifiers to update in an incremental fashion to accommodate new data without compromising classification performance on old data.

Learn++ [9], is an incremental learning algorithm inspired from AdaBoost and uses an ensemble of classifiers to achieve the task. Learn++ and AdaBoost generate an ensemble of weak classifiers, each trained with different distribution of training samples. The outputs of these classifiers are then combined using majority weighting scheme to obtain the classification rule.

The performance of the Learn++ depends on the number of classifiers specified that are to be generated. This value needs to be studied experimentally and put to use.

IV. Proposed Work

We aim to solve the cold start problem by improving adaptive clustering by proposing variations in adaptive clustering methods that will be used in the pre-processing stage to generate ratings, only for lowly ranked items so that they can compete with the highly ranked items.

We also develop a predictor based on incremental learning algorithm, LEARN++ that meets the needs of the adaptively clustered data. This is necessary since the adaptively clustered data may perform better in the initial stages of the system, but its performance deteriorates when the system reaches a stable state with each item having sufficient ratings without the use of adaptively clustered data. From this state onwards the need for adaptively clustered data reduces drastically and hence using an incremental learning algorithm would provide a smooth transition from cold start state to the ideal state of the system.

A. Pre-Processing Methods

The basic stages involve a clustering algorithm followed by a predictor model. In the training stage the data generated by the clustering algorithm is used to train the predictor model. One predictor model is allotted to each item to improve the prediction. In the testing stage the predictor is used to evaluate the ratings for unknown customer, item pairs. Based on this evaluation, the accuracy of the predictor is computed and analyzed.

B. Using Pearson's similarity measure with Adaptive Clustering

In this method, the process of adaptive clustering is associated with the Pearson's similarity measure to improve the approximation of ratings shared between items. The need for Pearson's similarity arises due to the fact that the ratings shared among the items do not have the same average rating. Suppose two items I1 and I2 have similar patterns of ratings among the customers that is, their correlation is high and item I1 shares a rating of item I2. Such items get adaptively clustered easily but the sharing of ratings without proper handling of the ratings can introduce higher errors in the performance of the predictors. Hence, **if we use methods like the Pearson's similarity** we can eliminate such errors to a large extent as the newly generated rating will be much closer

to the customer rating patterns for item I1 compared to that of item I2.

Pearson's correlation similarity(item I, item k)=

$(\sum_{j=1}^{L} (r_{ij} - ravg_i)(r_{kj} - ravg_j)) / \sqrt{(\sum_{j=1}^{L}(r_{ij} - ravg_j)^2)} \sqrt{(\sum_{j=1}^{L} (r_{kj} - ravg_k)^2)}$

Algorithm: Pearson's simalirity measure with Adaptive Clustering

Input: α ->minimum sum of ratings to form a cluster

Output: waiting list for items

1.cluster set C formed using k-means

2. Hash table H generated by adaptive clustering algorithm

Customer set a ϵ U

3.Perform Adaptive clustering as mentioned above.

4. **for** every item i in the hash table H **do**

5. List L= list of key i in hash table H

6. **for** every item k in L **do**

7. find correlation between i and k, Pearson's correlation similaty (i,k)

8. use correlation to find new ratings in i using the ratings of k

Using pred(i,a) = avg_rating(i)+ (Pearson's coefficient)(rating(k, a)- avg_rating(k))

9. add the new ratings to the ratings list of i.

10. **end for**

11. **end for**

C. Using Slope One predictor with Adaptive clustering

The original idea of Slope One predictors is simple and based on popularity differential between items for users. The use of slope one predictors can enhance the new ratings generated for items as generation of a new rating does not rely just on the rating of one item like in Pearson's similarity but on multiple items and multiple users to make a prediction.

Algorithms: Slope One predictor with Adaptive clustering

Input: item i ,customer c

L=list of all items that c has rated

D= list of top 50 customers who have rated majority of items in L

1.find deviation between item i and item j = $\sum_{(ui, uj)} (u_i - u_j)/ |S|$

(S is the list of common customers between i and j)

2. prediction(i, c) = $\sum_{j \epsilon L} (u_j + d_{i,j})/|L|$

Input: α ->minimum sum of ratings to form a cluster

1. cluster set C formed using k-means
2. Hash table H generated by adaptive clustering algorithm

Customer set a ϵ U

3. Perform Adaptive clustering as mentioned above.
4. **for** every i in the hash table H**do**

5. List L= list of key i in hash table H

6. find the list of new customers D for whom we need to generate ratings.

D=Φ

7. **for** each item k in L**do**

8. find new customers in item k that do not exist in i

D= D U {new list of customers from k}

9. **end for**

10. **for** each customer c in D

11. compute One Slope predictor(i, c) to generate the new rating

12. add new rating to list of ratings for item i

13. **end for**

14. **end for**

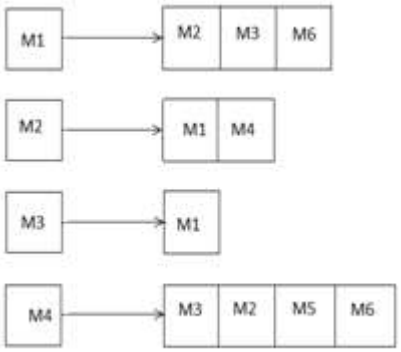

Fig. 1 Hash Table to maintain information of adaptively clustered items.

D. Incremental Learning Based Predictor

The standard predictor models are trained initially during the training stage and are used for evaluation/prediction over time. Using such predictors for adaptively clustered data may yield good results in the starting stages but over time the performance may drop as the predictor fails to take advantage of the incoming ratings provided by the user which could replace the adaptively clustered data for few items to improve their prediction performance. In order to overcome this issue, an incremental learning algorithm has been proposed to suit the requirements of adaptively clustered data. The incremental learning algorithm (LEARN++) uses an ensemble of weak classifiers such as a weakly trained neural network. In the initial stage the model is trained with the adaptively clustered data. Over time when sufficient ratings are received from the user for a particular item, we incorporate this new data into the algorithm by training a new weak classifier using this data. Each of the classifiers is associated with a weight that is used for majority voting to make the final prediction. The initially generated classifiers are assigned with value that is the ratio of non-adaptively generated ratings in the dataset. The subsequent classifiers are assigned the maximum weight in the ensemble of classifiers.

Algorithm: Incremental Learning

Initial training

1. **for** each item i
2. D=The training data is produced from the adaptive clustering stage.

B= Database block size

The training data is divided into databases with each database having a fixed number of training records.

3. **if** K databases are created, **then** K neural networks are generated.
4. Weight[i][k]= ratio of non-adaptively clustered data in the database for item i and classifier k

Incremental Learning stage

1. Majority_voting[5] initialized to 0.
2. **for** test record with item I **do**
3. **for** each classifier j **do**
4. rating=prediction of j for test record
5. Majority_voting[rating]=Majority_voting[rating]+weight of classifier j
6. **end for**
7. find rating with maximum vote and **return** the rating.
8. **for** each classifier j
9. **if** the prediction of j for test record matched majority rating **then do**
10. reward by increasing weight of classifier j by 0.5* present weight
11. **if** the prediction of j for test record does not match majority rating **then do**
12. punish by reducing weight of classifier j by 0.1*present weight
13. **end for**
14. store thistest record along with the rating provided by customer in D
15. **if** |D|=B **do**
16. generate a new classifier g
17. Weight[i][g]=the maximum weight of existing classifier
18. **end for**
19. add g to the existing list of classifiers.
20. **Or** Find classifier y in the ensemble of classifiers with minimum weight and replace y with g to form a new ensemble of classifiers.
21. **end for**

V. Experimental Results

The movie dataset was used to verify the performance of the proposed models. The MovieLens-100K data set is a classic movie rating data set collected through the MovieLens website. It consists of 100,000 ratings (1-5) from 943 users on1,682 movies and each user has rated at least 20movies. We propose to test our model using 80% of the data set to train the model and then use the remaining data to test the model.

K - Neareast Neighbor, Neural Network and proposed Incremental Learning algorithms are used as predictors to evaluate the performance on different pre processing methods as described earlier.

The commonly used standard evaluation measures are:

1. Mean Absolute Error:

MAE= $\sum_{i \in D}$ |PREDICTED RATING$_I$– ACTUAL RATING$_I$| / |D|

D = TEST DATA RECORDS

2. Root Mean Square Error:

RMSE= $\sqrt{(\sum_{i \in D}}$ (PREDICTED RATING$_I$ – ACTUAL RATING $_I)^2$ / $\sqrt{|D|}$

3. Precision:

P = |HITS| / |D|

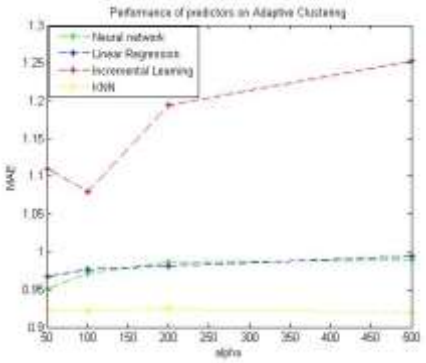

Fig. 2 Comparing predictors on Adaptive clustering

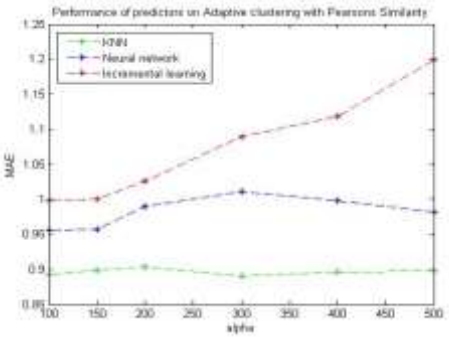

Fig. 3 Comparing predictors on Adaptive Clustering + Pearson's similarity

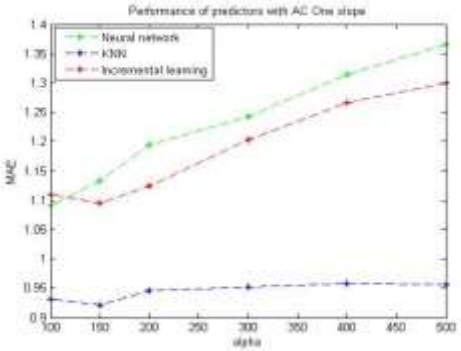

Fig. 4 Comparing predictors on Adaptive Clustering + One Slope

The adaptively clustered data is generated from the original training data for different values of α. The α values 50,100,200,500 were considered for observation. From the results (in Fig.2), it can be concluded that the KNN model outperforms all others models when the model is fed with adaptively clustered data. Both incremental learning and neural network models follow KNN. The incremental learning algorithm performs the worst compared to all the remaining models. Though, the models vary in performance with respect to each other in the adaptively clustered data, all the models show appreciable improvement compared to the each item and total clustered data. All the models tend to

achieve low error rates when α is between 100 and 150 ratings.

The best performance of KNN neighbor can be due to the fact the both the highly ranked items and lowly ranked items have similar number of ratings. As a result, the performance of the movies in the tail improves while not disturbing the performance of movies in the head. The incremental learning algorithm has poor results due to the unavailability of sufficient data in the test set. Due to unavailability of sufficient data, very few new classifiers are generated for each movie that do not bring much improvement in the results. If the incremental learning model is provided with large amount of data, then over time the model may converge to yield the best results possible.

In order to bring further improvements in adaptively clustered data, we try to enhance the newly generated ratings for a movie to approximate the closest value prediction to the actual rating provided by user. To bring such an improvement, the Pearson's similarity measure is applied during the adaptive clustering process before generating the new rating for movies in the tail. The generated data is trained and tested on the previously mentioned predictors.

To study this experiment different alpha values of 50, 100, 150, 200, 300, 400, 500 are considered. The Pearson's similarity improves the error rates for all the predictors compared to the normal adaptive clustering method (Fig.3). The KNN model has the least error of all models again and this time, both the KNN model and the neural network model perform better than the collaborative filtering models by showing lower performance errors than the benchmark models.

In Figure 4, the adaptive clustering is paired with Slope One predictor. To study this experiment different alpha values of 50, 100, 150, 200, 300, 400, 500 are considered. The Slope One predictor improves the error rates for all the predictors compared to the normal adaptive clustering method. The KNN model has the least error of all models again. The results are comparable to the Pearson's similarity methods but for α greater than 150 the error for the neural network and incremental learning methods grows rapidly. This may be due to the fact that for larger values of α the one slope predictor uses ratings of large number of movies resulting in consideration of ratings of a large number of users disrupting the balance for error.

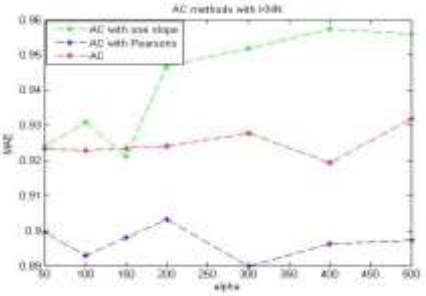

Fig. 5 Performance of KNN on different variations of Adaptive clustering

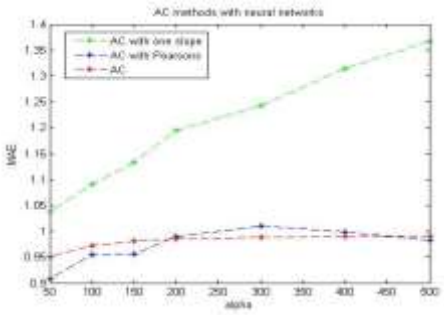

Fig. 6 Performance of neural network on variations of adaptive clustering

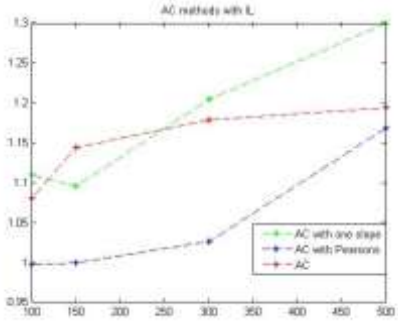

Fig. 7 Performance of Incremental learning algorithm on variations of adaptive clustering

The KNN model gives the best results compared to other predictor models like neural networks and incremental learning methods. All the clustering methods show optimal performance for α between 100 and 150. The adaptive clustering method with Pearson's similarity outweighs the performance of other clustering methods. The normal adaptive clustering method seems to perform comparatively better than the AC with Slope One method.

Both simple AC and AC with Pearson's similarity perform well on the neural network and attain low error rates for α between 100 and 150. Though, the performance of AC with Slope One is not as great as the other methods, the error is comparatively lower than total clustering and each item method[3].

In the increment learning method, the AC method with Pearson's similarity gives the best performance followed by the AC method with Slope One predictor. All the AC methods achieve low error rate for α between 100 and 150 ratings. The drastic increase in error for Slope One may be due to large value of α resulting in interaction with ratings from large number of users to generate a rating.

VI. Results

In order to solve the cold start problem in items, various adaptive clustering improvements were proposed and tested for performance. Two improvements are suggested to improve the adaptive clustering process ie. Adaptive clustering with Pearson's similarity and Adaptive clustering with Slope One predictor. Each of these methods try to improve the existing

clustering process and reduce the error rates in the lowly ranked items.

An incremental learning algorithm also has been proposed to suit the needs of adaptive clustering. Using this algorithm, the weightage of adaptively clustered data can be reduced over time and the system can make a smooth transition from pre-processed data towards the true data that is provided by the customers.

A study of the improvement of error performance of the proposed methods has been done and tabulated as follows:

TABLE I
STUDY OF THE IMPROVEMENT OF ERROR PERFORMANCE OF THE PROPOSED METHODS

Methods	Performance improvement
AC with Pearson's on Each item method	23.01%
AC with Pearson's on total clustering method	18.91%
AC with Pearson's on adaptive clustering method	5.10%
AC with Slope One on Each Item Method	15.93%
AC with Slope One on total clustering method	14.3%
AC with Slope One on adaptive clustering method	3.2%

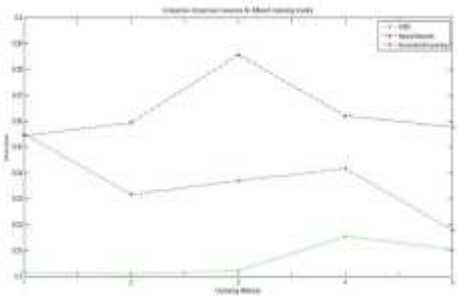

Fig. 8 Evaluating precision of different clustering models

The clustering methods in the Fig.8 are numbered as follows:

1. Each Item Method
2. Total clustering
3. Adaptive Clustering
4. Adaptive Clustering with Pearson's similarity
5. Adaptive Clustering with Slope One

From Fig.8, it can be observed that the neural network provides the highest precision possible. The precision rates of KNN are the least and that of incremental learning method lie in intermediate.

The neural network attains high precision for adaptive clustering method whereas the remaining predictors attain high accuracy for adaptive clustering with Pearson's similarity method. Neural network achieves a highest precision of 38.57% compared to 31.55% in KNN and 35.52% in incremental learning method.

VII. CONCLUSION

In order to solve the cold start problem, improvements have been suggested for the adaptive clustering process that show considerable improvement in the error rates of the lowly ranked items. An incremental learning predictor is also proposed that learns from time to time and incorporates the learning of new information. The study of adaptive clustering is limited to the selection of α. The future work could involve work on dynamic change of α for appropriate clustering of data arriving from data streams.

REFERENCES

[1] Namita Mittal, RichiNayak. MC Govil and KC Jain," Recommender System Framework using Clustering and Collaborative Filtering", *Third International Conference on Emerging Trends in Engineering and Technology*,IEEE,2010.

[2] Gediminas Adomavicius and Alexander Tuzhilin," Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions", *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, VOL. 17, NO. 6, JUNE 2005,pp, 734-749.

[3] Yoon-Joo Park," The Adaptive Clustering Method for the Long Tail Problem of Recommender Systems", *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, VOL. 25, NO. 8, AUGUST 2013,pp. 1904-1915.

[4] Ling Yanxiang, Guo Deke and CaiFei , Chen Honghui," User-based Clustering with Top-N Recommendation on Cold-Start Problem"**,** *Third International Conference on Intelligent System Design and Engineering Applications,IEEE*, 2013,pp. 1585-1589.

[5] Y.J.Park and A.Tuzhilin, "The Long Tail of Recommender Systems and How to Leverage It," *Proc. ACM Conf. Recommender Sytems, pp. 11-18,2008*

[6] Ismail SengorAltingovde, ÖzlemNurcanSubakan and ÖzgürUlusoy," Cluster searching strategies for collaborative recommendation systems", *I.S. Altingovde et al. / Information Processing and Management 49*,2013,pp. 688-697.

[7] Hridya Sobhanam and A.K.Mariappan,"A Hybrid Approach to Solve Cold Start Problem in Recommender Systems using Association Rules and Clustering Technique**",** *International Journal of Computer Applications (0975 – 8887)*, Volume 74– No.4, July 2013,pp. 17-23

[8] J. Han and M. Kamber. 2000. Data mining: Concepts and Techniques.Morgan-Kaufman, New York

[9] Robi Polikar,LalitaUdpa, Satish S. Udpa,," Learn++: An Incremental Learning Algorithm forSupervised Neural Networks**",** *IEEE transactions on Systems , Man and Cybernetics. Vol.31,pp. 1094-1106*

[10] L.Ungar and D.Foster, "Clustering methods for collaborative filtering", *Proceedings of the workshop on Recommendation Systems,AAAI Technical Report WS-98-09,AAAI Press,Menlo Park,CA,1998,pp,114-129.*