IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 16, 2017, accepted March 13, 2018, date of publication March 26, 2018, date of current version May 2, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2819428

Hybrid Real-Time Matrix Factorization for Implicit Feedback Recommendation Systems

CHIA-YU LIN, (Student Member, IEEE), LI-CHUN WANG, (Fellow, IEEE), AND KUN-HUNG TSAI

Department of Electrical and Computer Engineering, National Chiao Tung University, Hsinchu 30010, Taiwan

Corresponding author: Li-Chun Wang (lichun@cc.nctu.edu.tw)

ABSTRACT In this paper, we present a hybrid real-time incremental stochastic gradient descent (RI-SGD) updating technique for implicit feedback matrix factorization (MF) recommendation systems. Compared with explicit feedback evaluation scores, implicit feedback data are easier to obtain but pose challenges to MF recommendation systems because of the transformation procedures from raw data to user preference scores. Another challenge for MF recommendation systems is the accuracy issue when the speed of the new input data increases. The proposed RI-SGD is designed for computationally-efficient and accurate time-variant implicit feedback MF recommendation system, which consists of alternating least squares with weight regularization in the training phase and stochastic gradient descent in the updating phase. To demonstrate the advantages of the RI-SGD updating technique in terms of computational efficiency and accuracy, we implement the proposed updating techniques in a real-time music recommendation system. Compared with the method of retraining the entire model, our numerical results show that RI-SGD approach can achieve almost the same recommendation accuracy, but requires only about 0.02% of the retraining time.

INDEX TERMS Recommendation system, implicit feedback, matrix factorization, real-time updating.

I. INTRODUCTION

Recommendation systems analyze customers' behavior to predict products of interest for potential customers. The leaders of e-commerce, such as Amazon, e-Bay, Netflix, and Youtube etc., have made their websites play the role of personalized recommenders, which has influenced consumer buying behavior significantly. Collaborative filtering (CF) can recommend complex items to users without explicit files of users and items. Consequently, CF becomes appealing recommendation techniques [1], [2]. For example, Netflix adopts CF to predict user ratings for films [3].

Neighborhood methods and latent factor models are two major collaborative filtering techniques [4], [5]. Neighborhood methods are simple and intuitive, which analyze the similarities between users or items. Some latent factor models apply matrix factorization to discover the hidden features between items and users from the raw data of customers' ratings. High correspondence for certain features between items and users leads to recommendation. Because of the advantages of preserving accuracy for scaling data, low computational cost and reducing the problem from high levels of sparsity, matrix factorization latent factor models have received a great deal of attention recently [6]–[8].

However, most current matrix factorization recommendation techniques are executed in batch mode based on the whole data stored in database. Thus, this kind of matrix factorization recommendation systems in batch mode are trained in a single pass and ignore the issue of input data update. In practice, however, new users and items frequently enter the recommendation system. If the new input data are not used to update model, the original recommendation model built in the batch mode may become obsolete, causing poor recommendations.

Different types of input data are another important design issue for a recommendation system. Current matrix factorization methods are mostly designed with explicit feedback data, which are easy to analyze. However, service providers have to design feedback pages for enquiring customers rating, which is time consuming and hard to collect data due to user interventions. On the contrary, implicit feedback data such as browsing history and search patterns can be easily collected via a more natural way. Nevertheless, a recommendation system with implicit feedback needs to tackle the issues to map implicit feedback to user's preference scores [9]. First, there is no negative feedback of implicit feedback. Secondly, implicit feedback is inherently noisy and hard to

transfer non-negative confidence value into useful preference value. Thirdly, appropriate measuring the evaluation of implicit feedback recommender is hard. Therefore, matrix factorization should be modified to adopt implicit feedback data.

In the literature, matrix factorization recommendation approaches are investigated from two aspects:

- From the aspect of implicit feedback, [9] revised the original cost function to overcome implicit feedback problem. In [10], different kernel functions were suggested to improve the recommendation accuracy in the process of analyzing implicit feedback. For top-n recommendation with implicit feedback, [11] proposed local weighted matrix factorization (LWMF). LWMF employed the kernel function to intensify the local property of sub-matrices and the weight function to model user preferences. Although [9]–[11] can achieve high accuracy, the issue of real-time updating model has not been considered yet.
- On the other hand, from the aspect of the real-time updating issue, [12] proposed a real-time updating method using stochastic gradient descent (SGD), but only for explicit feedback recommendation. Vinagre *et al.* [13] solved a real-time updating problem with binary value of implicit feedback, but the authors directly changed the implicit feedback to binary values without adopting the real numerical implicit feedback value. Thus, the accuracy of the algorithm in [13] is not high. Reference [14] designed a learning algorithm to efficiently optimize a MF model with variable-weighted missing data based on element-wise alternating least squares (eALS) technique. Reference [15] proposed one-sided least squares (One-sided LS) to integrate with ALS with faster learning speed. However, ALS suffers from worse running performance than SGD when considering the updating process of a single element [16], [17], which results in longer delay in real-time applications.

Therefore, we integrate the advantages of ALS and SGD into a hybrid real-time incremental stochastic gradient descent (RI-SGD) updating technique for implicit feedback MF recommendation systems. To evaluate the performance of RI-SGD and other updating algorithms, we conduct experiments on IBM Streams, which is a real-time streaming data analytic platform developed by IBM [18]–[20]. Our results show that RI-SGD can achieve competitive accuracy in comparison with all updating methods and only consume 0.02% of updating time of retraining entire model. A light-weight adaptive updating concept was proposed in our previously published work [21]. Based on the previous work, we further propose the architecture of RI-SGD recommendation system, time-dependent cost function and incrementally updating model. Moreover, we discuss the effects of number of recommendation items N and attenuation parameter α, which will be detailed in Section VI. We also use normalized discounted cumulative gain (NDCG) to show the recommendation quality of RI-SGD in comparison with other different updating methodologies.

The rest of this paper is organized as follows. Section II introduces background and related work of implicit feedback matrix factorization recommendation system. Section III formulates the updating problem in recommendation systems. Section IV presents the RI-SGD updating algorithm for implicit feedback MF recommendation systems. Section V shows RI-SGD recommendation system. The experiments and numerical results are demonstrated in Section VI. Finally, we give our concluding remarks in Section VII.

II. BACKGROUND AND RELATED WORK

In this section, we introduce matrix factorization and how matrix factorization do the recommendation for implicit feedback.

A. MATRIX FACTORIZATION APPROACH

Compared with similarity-based recommendation method, which considers the similarity between users and items to make recommendation, matrix factorization recommendation method is more memory-efficient and more accurate [22]. In the well-known Netflix competition [23], matrix factorization method was adopted to deal with the rating prediction issue [5]. To recommend N items with f latent factors to M customers, the memory cost of matrix factorization recommendation system is $f * M + f * N$, but the similarity-based recommendation method has the memory cost of $M^2/2$.

Similar to singular value decomposition (SVD), matrix factorization decomposes the user-item record matrix into multiplication of two low-rank matrices, in which the latent factors of users and items can be stored. Thus, matrix factorization can discover items with similar contents as well as implicit features. As shown in Fig. 1, the $M \times N$ rating matrix Γ is decomposed into the user latent factor matrix X and the item latent factor matrix Y. The factor vectors include the attributes of items, such as music types, movie genre, tags in the twitter message, etc. A high score of a customer's factor means that a customer prefers to that particular attribute. The matrix factorization recommendation system for user u and item i will result in a user vector $x_u \in \mathbf{R^f}$ and the item vector $y_i \in \mathbf{R^f}$, respectively. $x_u^T y_i$ captures the interaction between user u and item i as shown in Fig. 2. In the figure, Bob has a user vector [0.4, 0.6] on two different factors (dim1, dim2). Beatles and Utada are two artists. They also have different values in dim1 and dim2. The resulting dot product $x_u^T y_i$

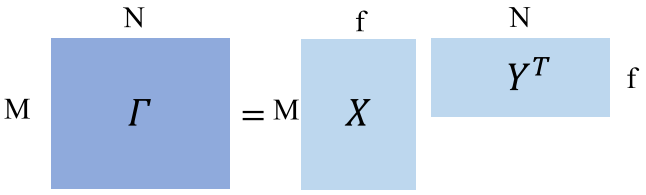

FIGURE 1. The concept of the matrix factorization approach with M customers, N items, f latent features.

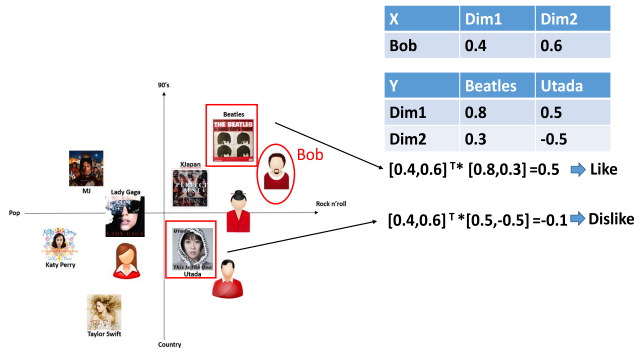

FIGURE 2. Illustration of the latent feature of users and items.

shows the Bob's preference for two artists. One can observe that if a user is in favor of a specific artist, the user will have a similar location in the space.

The major challenge for matrix factorization is to find the mapping between users and items in factor vectors. To learn the factor vectors x_u and y_i, the value in rating matrix Γ is used as the training data and is decomposed into $x_u^T y_i$ with the objective of minimizing the following cost function:

$$\underset{x^*,y^*}{\text{minimize}} \sum (r_{u,i} - x_u^T y_i)^2 + \lambda(\|x_u\|^2 + \|y_i\|^2), \quad (1)$$

where $r_{u,i}$ is the rating of user u on item i and λ is used for regularizing the recommendation model to avoid overfitting [5].

The most successful methods to minimize (1) are Stochastic Gradient Descent (SGD) [16] and Alternating Least Squares (ALS) [24]. SGD predicts r_{ui} and computes the associated prediction error for each training case. Then SGD updates the corresponding x_u and y_i by the inverse direction of the gradient of the error. ALS rotates between fixing the $x_u's$ and $y_i's$ to make (1) become quadratic and can be solved optimally. As discussion in [16] and [17], SGD combines implementation ease with a relatively fast running time and higher model accuracy than ALS especially using sparse data.

B. IMPLICIT FEEDBACK MATRIX FACTORIZATION RECOMMENDATION SYSTEM

The implicit feedback data, such as transactions and playlist records, are automatically collected by a recommendation system without the intervention of customers' direct reactions. Take music recommendation as an example. If a customer repeatedly listens to a song several times, this customer may like this music style or singer. The number of listening times can be implicit feedback data for recommendation system. However, the frequency of playing a certain song is non-negative value and has a very large dynamic range, which cannot be easily applied for rating a customer's preference.

On the contrary, like questionnaires and rankings, explicit feedback has fixed scale rating, which can easily reflect a customer's preference. However, because of the complex collecting procedures, the amount of explicit feedback is much less than that of implicit feedback. Because of abundant implicit feedback training data, more and more real-time recommendation systems adopt implicit feedback rather than explicit feedback.

Since the value of implicit feedback cannot be directly mapped to users' preferences, an explicit feedback matrix factorization recommendation system shall be modified, and this is not a straightforward task. Hu *et al.* [9] modified the cost function of matrix factorization for implicit feedback data. They replaced the original rating r_{ui} in (1) with the binary value p_{ui}, and the confidence weighting c_{ui} as follows:

$$\underset{x^*,y^*}{\text{minimize}} \sum c_{ui}(p_{ui} - x_u^T y_i)^2 + \lambda(\|x_u\|^2 + \|y_i\|^2), \quad (2)$$

where $c_{ui} = (1 + \alpha r_{ui})$; α is the attenuation parameter for implicit feedback; r_{ui} is the rating value of explicit feedback; and p_{ui} is derived by binarizing r_{ui} which indicates the user's preference on the rating item. The values of p_{ui} are

$$p_{ui} = \begin{cases} 1, & r_{ui} > 0 \\ 0, & r_{ui} = 0. \end{cases} \quad (3)$$

The cost function contains $M \cdot N$ terms, which can easily reach a few billion. That is, the training data are not sparse. Looping over each single training case as SGD does would not be practical. ALS can efficiently deal with implicit feedback data. Therefore, [9] proposed an alternative least square with weight regularization (ALSWR) to solve the revised cost function with implicit feedback (2). Rendle and Thieme [10] introduced the regularized kernel matrix factorization models to cope with implicit feedback data, in which the prediction equation with different kernel functions was used for accomplishing non-linear mapping and confining the range of predicted values. In [11], Wang *et al.* decomposed the sub-matrix instead of the original matrix to increase the computation efficiency of matrix factorization with implicit feedback and to relieve the sparsity problem. To model user preference, the authors adopted the kernel function and proposed local weighted matrix factorization (LWMF), which combined local low-rank matrix approximation (LLORMA) with weighted matrix factorization (WMF) to strengthen local property and the weight function.

III. UPDATING PROBLEMS IN RECOMMENDATION SYSTEMS

In the real world applications, user feedback is continuously being generated. To maintain recommendation accuracy, an on-line updating model should be performed. However, most of the existing matrix factorization recommendation systems are implemented in a batch mode, thereby requiring to retrain the entire model with new incoming data [9], [10]. It is time consuming to retrain the entire recommendation model. Hence, it is essential to develop a computationally efficient updating mechanism for real-time matrix factorization recommendation system.

About the literature of real-time updating techniques for explicit feedback matrix factorization recommendation system, Ling *et al.* [12] proposed the explicit rating-oriented

matrix factorization (PMF) and ranking-oriented matrix factorization (RMF) to incrementally handle new rating values without retraining the models. The authors also showed that PMF and RMF algorithms were linearly scaled with the number of observed ratings, and had comparable performance as the batch-trained algorithm. Reference [25] and [26] developed selecting outdated information and forgetting such information to achieve better accuracy of recommendation. However, these techniques cannot deal with subtle changes of feedback data such as individual user's preference. Reference [27] designed a linear feature transformation of user and item latent vectors over time to generalize incremental MF framework for explicit feedback.

Now for implicit feedback, Vinagre *et al.* [13] introduced an incremental SGD algorithm (ISGD) to transform implicit records to a binary rating matrix, as shown in Fig. 3. Because binary rating matrix disregards the real value, the accuracy of such a recommendation system is questionable. Reference [13] proposed a learning algorithm based on the element-wise alternating least squares (eALS) technique to efficiently optimize a MF model with variably weighted missing data. Reference [15] designed one-sided least squares (One-sided LS) to reduce compute and storage cost of incremental learning of MF model. However, when updating a single element, the computation process of ALS is much more complex than SGD and suffers from poor running performance.

	I1	I2	I3	I4	I5
U1	10	20			5
U2		1	3		33
U3	3				
U4		18		1	

⇨

	I1	I2	I3	I4	I5
U1	1	1	0	0	1
U2	0	1	1	0	1
U3	1	0	0	0	0
U4	0	1	0	1	0

FIGURE 3. Replace the rating matrix element with binary value.

IV. THE PROPOSED HYBRID RI-SGD ALGORITHM

Implicit feedback data are continuously generated but each data is only related to one user and one item. Although ALSWR is suitable for matrix factorization with implicit feedback, it still suffers from much complex and worse running performance than SGD when updating the prediction value for one user and one item. Therefore, ALSWR and SGD are integrated into a hybrid real-time incremental stochastic gradient descent (RI-SGD) MF recommendation system with implicit feedback. RI-SGD consists of three key elements: 1) training model, 2) time-dependent cost function, and 3) incrementally updating model.

A. TRAINING MODEL

The goal in the training process is to obtain $x_u \in \mathbf{R^f}$, and $y_i \in \mathbf{R^f}$ to minimize the cost function (2). To efficiently build a matrix factorization model for few billion users and items, we adopt ALSWR [9] to train the model. ALSWR follows that

$$x_u = (Y^T C^u Y + \lambda I)^{-1} Y^T C^u p(u) \tag{4}$$

and

$$y_i = (X^T C^i X + \lambda I)^{-1} X^T C^i p(i), \tag{5}$$

where all user-factors are arranged within $\mathbf{M} * \mathbf{f}$ matrix X and all item-factors are gathered within $\mathbf{N} * \mathbf{f}$ matrix Y; $C^u \in \mathbf{R^{N*N}}$ and $C^i \in \mathbf{R^{M*M}}$ are the diagonal matrix with $C^u_{ii} = c_{ui}$ for user u and that with $C^i_{uu} = c_{ui}$ for item i, respectively; $p(u) \in \mathbf{R^M}$ and $p(i) \in \mathbf{R^N}$ respectively represent the binary confidence vector for user u and that for item i. From x^u and y_i, we can find the largest value of $x_u^T y_i$ and recommend item i to user u.

B. TIME-DEPENDENT COST FUNCTION

As the new data $r_{u,i}$ of user u on item i enter the system at time t, the cost function (2) can be re-write as

$$\underset{x^*,y^*}{\text{minimize}} \sum c_{ui}(t)(p_{ui}(t) - ((x_u(t))^T y_i(t))^2 + \lambda(\|x_u(t)\|^2 + \|y_i(t)\|^2), \tag{6}$$

where $c_{ui}(t) = (1 + \alpha r_{ui}(t))$; α is the attenuation parameter for implicit feedback; $p_{ui}(t) = 1$ if $r_{ui}(t) > 0$, otherwise, $p_{ui}(t) = 0$.

C. INCREMENTALLY UPDATING MODEL

After modifying $c_{u,i}(t)$ and $p_{u,i}(t)$, we need to deal with the latent feature vectors. The ALSWR algorithm needs to retrain the entire model based on the new entries, resulting in very long delay. For small changes in rating matrix Γ, it is unnecessary to recalculate the entire model. Take a 4×5 rating matrix as an example, which is expressed as

$$\begin{pmatrix} r_{11} & \cdots & \mathbf{r_{15}} \\ \vdots & \ddots & \vdots \\ r_{41} & \cdots & r_{45} \end{pmatrix} = \begin{pmatrix} \mathbf{x_{11}} & \cdots & \mathbf{x_{1f}} \\ \vdots & \ddots & \vdots \\ x_{41} & \cdots & x_{4f} \end{pmatrix} \begin{pmatrix} y_{11} & \cdots & \mathbf{y_{15}} \\ \vdots & \ddots & \vdots \\ y_{f1} & \cdots & \mathbf{y_{f5}} \end{pmatrix}. \tag{7}$$

If $r_{1,5}$ changes, we only need to update corresponding two vectors $X_{1,f}$ and $Y_{f,5}$.

SGD performs relatively fast when considering the calculation for one user and item. Therefore, incremental SGD is proposed to update the matrix factorization recommendation model. We calculate the first order gradient of the cost function (6). Then we obtain

$$\frac{\partial C(x,y)}{\partial x_u} = \gamma(c_{u,i}(t)(p_{u,i}(t) - x_u^T y_i)y_i) + 2\lambda x_u), \tag{8}$$

and

$$\frac{\partial C(x,y)}{\partial y_i} = \gamma(c_{u,i}(t)(p_{u,i}(t) - x_u^T y_i)x_u) + 2\lambda y_i). \tag{9}$$

Next we update the corresponding latent feature vectors of user u and item i as follows:

$$x_u \leftarrow x_u - \gamma(c_{u,i}(t)(p_{u,i}(t) - x_u^T y_i)y_i) + 2\lambda x_u), \tag{10}$$

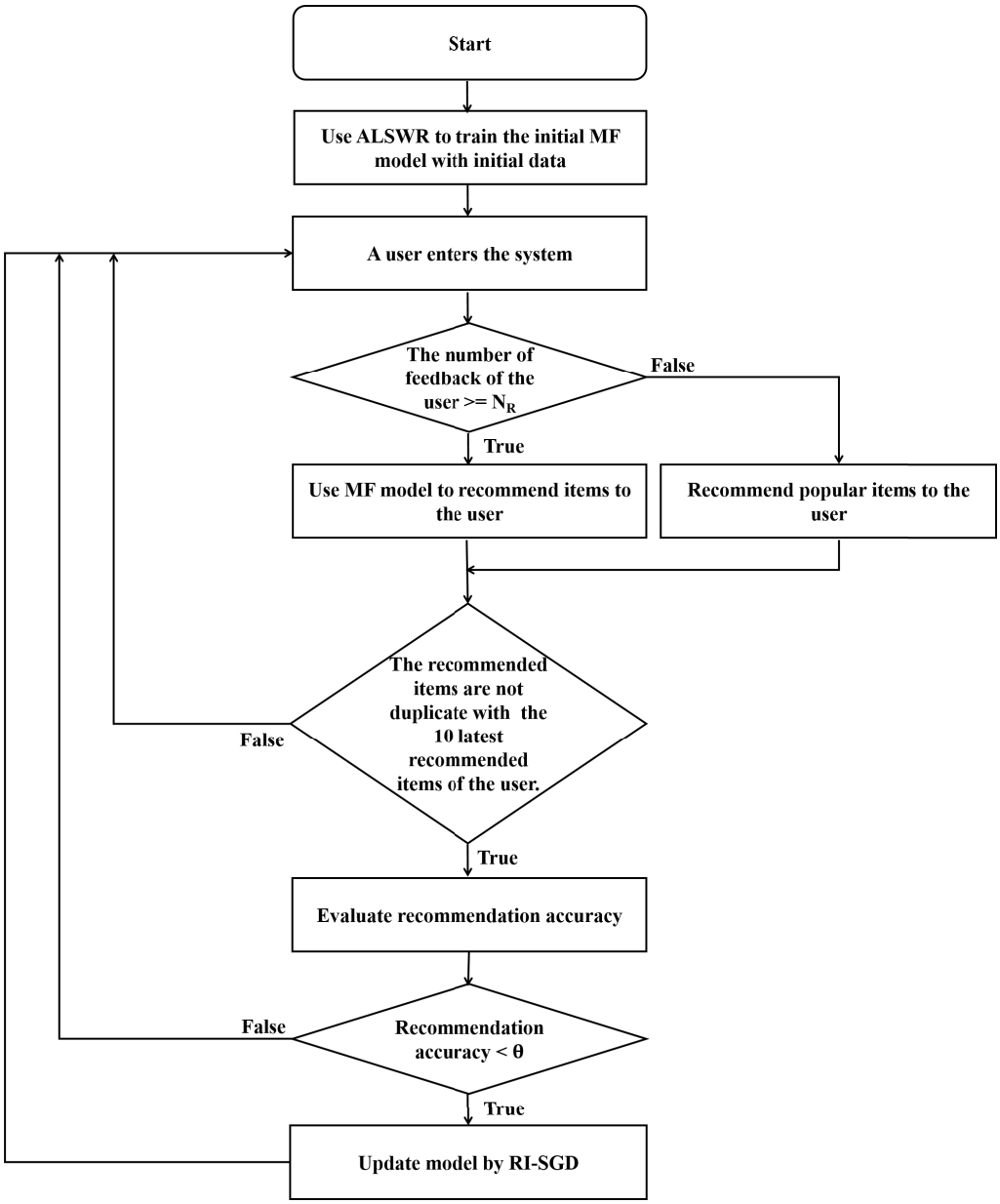

FIGURE 4. Procedures for model updating.

and

$$y_i \leftarrow y_i - \gamma(c_{u,i}(t)(p_{u,i}(t) - x_u^T y_i)x_u) + 2\lambda y_i). \quad (11)$$

Denote Ω as the number of non-zero observations. The updating cost by using our method is $O(|\Omega|f)$. However, the cost of retraining the entire model $O(|\Omega|f^2 + (M+N)f^3)$. As such, our method can reduce updating complexity significantly.

Fig. 4 shows the updating procedure of RI-SGD. We train the initial MF model by ALSWR. When a user enters the system, the system counts the total number of feedback data of the user. If the total number of feedback data of the user is larger than a threshold (N_R), the MF model is adopted to recommend items to the user. When having few records or a new user in the system, the current training model cannot make a good recommendation. In this case, the current popular items are recommended to the user. The recommendation accuracy of recommending different items is measured to judge whether the model needs to be updated. If the recommended items are totally similar with ten latest recommended items, the accuracy is not measured. Otherwise, we evaluate recommendation accuracy as

$$Accuracy = hitcount/number\ of\ records, \quad (12)$$

which is widely used in evaluating top-N recommendation systems [28]. Hitcount is plus 1 when the user chooses the recommended item. If the recommendation accuracy is

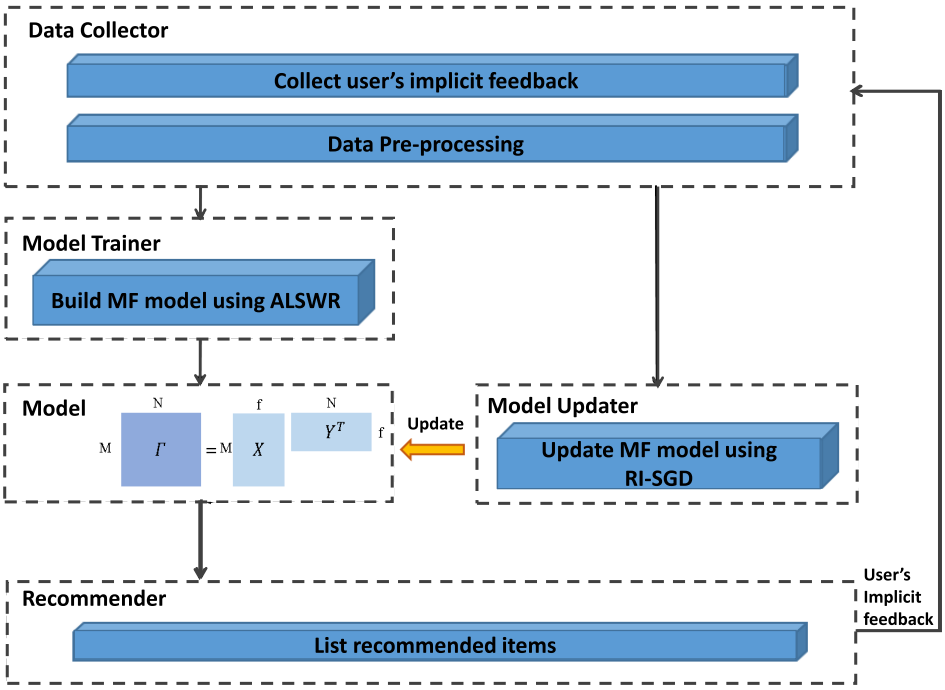

FIGURE 5. Framework of implicit feedback MF recommendation system with RI-SGD.

smaller than a threshold θ, the RI-SGD updating process is activated to update the MF model.

V. RI-SGD RECOMMENDATION SYSTEM

We implement an RI-SGD recommendation system on IBM Streams, which is a real-time streaming data analytic platform developed by IBM [18]–[20]. As shown in Fig. 5, the considered recommendation system consists of four modules: 1) data collector, 2) model builder, 3) model updater, and 4) real-time recommender.

- Data Collector: In this model, implicit feedback data with timestamp from different sources are collected and pre-processed by the operators of IBM Streams, such as filters, sort, de-duplicate and join.
- Model Trainer: On a Java operator of IBM Streams with Apache Mahout[1] [29], the ALSWR algorithm for matrix factorization is implemented to train the latent factor model.
- Model Updater: The proposed RI-SGD approach is implemented as a Java operator to update the latent factor model.
- Real-time Recommender: When receiving recommendation requests, the system checks the number of records associated with this customer. If the number of records is smaller than the predefined threshold, the system will recommend popular items among other users, rather than predicting customer's preference. While the number of records is enough, the latent vectors x_u and y_i will be calculated for $i = 1, \ldots N$. The inner product of $x_u^T y_i$ are sorted in a descending order, from which the top list of items are recommended to customer u. Customer's implicit feedback to the recommended items such as buying the recommended items are recorded and sent to the data collector for model updating. The recommendation accuracy is also measured to determine whether it is necessary to update the current model.

[1]A project sponsored by Apache and contains several classical recommendation algorithms including neighborhood-based method and matrix factorization using ALSWR.

VI. EXPERIMENTS AND DISCUSSIONS

In this section, we evaluate the accuracy of different number of recommendation items N and different attenuation parameter α to decide the most appropriate N and α for the following experiments. We also design experiments to compare the updating time, the recommendation accuracy and the normalized discounted cumulative gain (NDCG) of seven updating methodologies.

A. DATASET PREPARATION

We perform experiments based on Last.fm [30], [31] to evaluate the performance of the proposed RI-SGD algorithm. The dataset contains 19 million music ordering records of nearly 1,000 users, each of which includes user id, timestamp, artist id, artist name, track id and track name.

To avoid bias, data are pre-processed. Artist id and track id are deleted in data since there are not artist id and track id in some records. Also, all the songs of an artist are combined into one record to avoid the records with low ordering rate. However, there is still 30% of artists were chosen only one

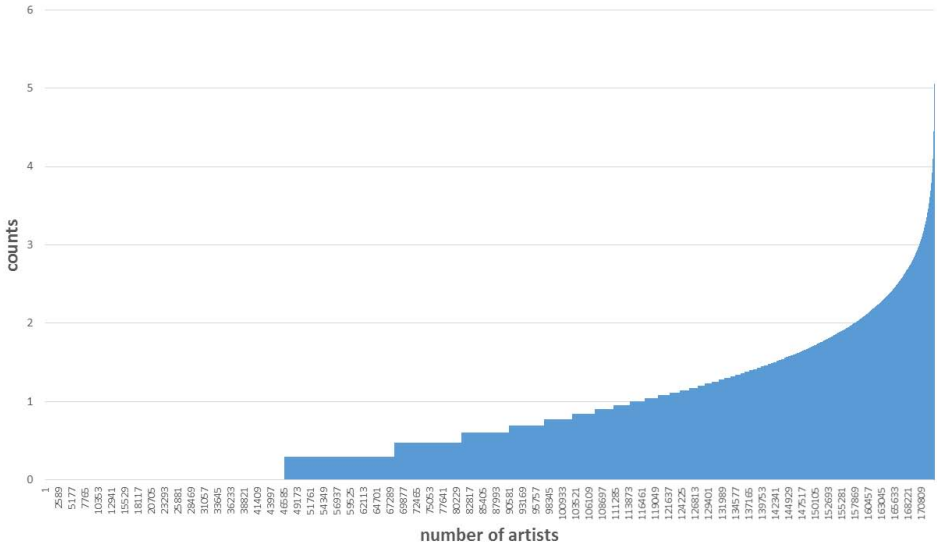

FIGURE 6. Histogram of artist count (log) to number of users.

time as shown in Fig. 6. Therefore, 50,000 records ordered by only one time are deleted. Then, the data is sorted by timestamps and split into training data and testing data. In Case 1, we randomly select 5% of dataset for training and the rest of data are testing data. In Case 2, 10% of dataset are trained and the rest of data are tested.

B. METHODOLOGY AND EVALUATION

In our experiments, we develop RI-SGD based on Apache Mahout [29] in IBM Streams [20] to show that RI-SGD can intensively reduce the updating time and keep high accuracy.

As shown in Fig. 4, we use ALSWR algorithm to train an initial matrix factorization model by 5% and 10% of the data. The rest 95% and 90% of data will be sent to the system one by one to simulate continuously streaming data. The number of latent factor vectors $f = 20$ is set since most of the information we need in the model can be preserved. Regularization parameter λ is set to 0.1. When a user u enters the system, the ordering times of an artist represents the rating of the user on item i as $r_{u,i}$. The confidence weighting is $c_{ui} = (1 + \alpha r_{ui})$. The system recommends artists to the user and uses (12) to measure the recommendation accuracy. If the recommendation accuracy is smaller than θ, the updating process is activated with the following different updating methodologies.

- The model is not updated. We call the method "old" in the following experiments.
- In the updating process, we use ALSWR to retrain the entire model. The method is called "ALSWR" in the following experiments.
- We retrain the entire model by ALSWR with window. The window contains latest 5% and 10% of data. "ALSWR window" is the abbreviation in the following experiments.
- We update model by binary SGD proposed in [13]. We call it "ISGD" in the following experiments.
- We update model with the most popular items among all users. The abbreviation is "pop" in the following experiments.
- We update model by the proposed RI-SGD. The abbreviation is "RI-SGD" in the following experiments.
- To evaluate the performance of limiting range of predicted value. Cost function (2) is combined with logistic function

$$K_s(x_u, y_i) = \phi_s(b_{x,y} + \langle x_{ui}, y_i \rangle). \tag{13}$$

We can obtain kernel cost function as

$$(1 + exp(c_{ui}(p_{u,i} - x_u^T y_i)^2 + \lambda(\|x_u\|^2 + \|y_i\|^2))^{-1}. \tag{14}$$

The proposed RI-SGD is used to update kernel cost function. It is called "RI-SGD log" in the following experiments.

The setting of θ, number of recommendation items N and attenuation parameter α will be discussed in next subsections.

C. NUMERICAL RESULTS AND OBSERVATION

1) NUMBER OF RECOMMENDATION ITEMS N

Figs. 7 and 8 show the recommendation accuracy (12) for $N = 5, 10, 20$. We find that the average accuracy increases with a large value of N. Because the number of recommended items is smaller than 15 in most recommendation applications [28], [32], N is set to 10. Therefore, the recommendation accuracy should be larger than 10%. Thus, the accuracy threshold $\theta = 0.1$, which is used to decide whether the model needs to be updated.

2) ATTENUATION PARAMETER α

In the cost function of implicit feedback matrix factorization (2), the attenuation parameter α in the confidence weighting $c_{ui} = (1 + \alpha r_{ui})$ decides how the implicit value

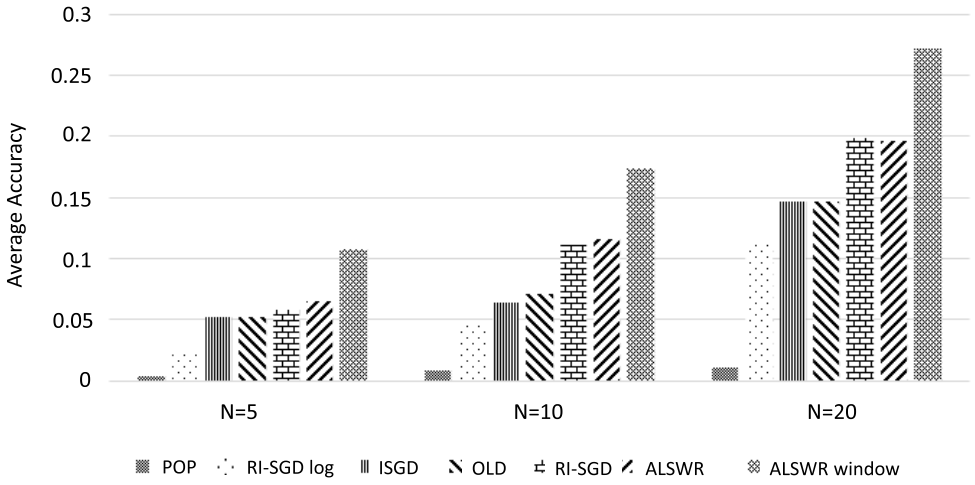

FIGURE 7. The accuracy of different N in Case I.

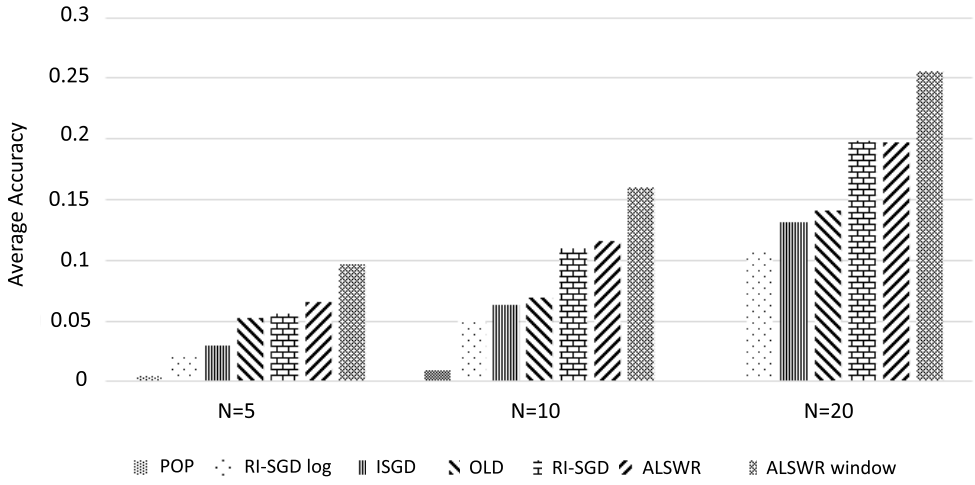

FIGURE 8. The accuracy of different N in Case II.

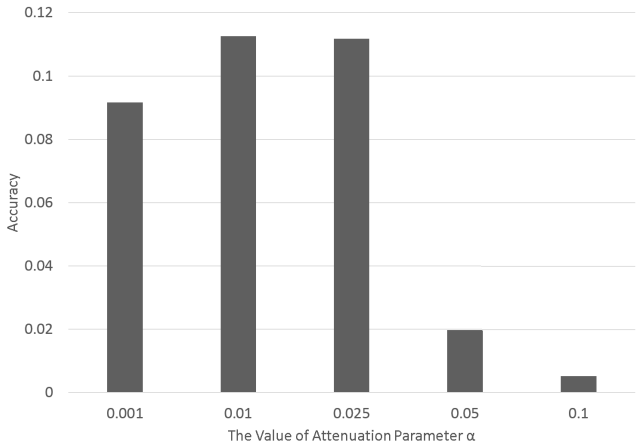

FIGURE 9. The accuracy of different attenuation parameter α in Case I.

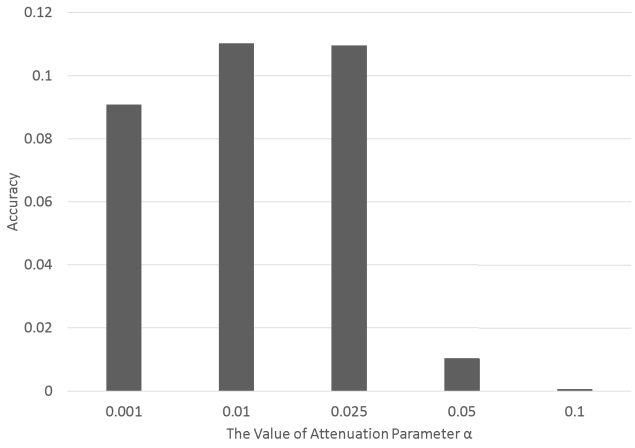

FIGURE 10. The accuracy of different attenuation parameter α in Case II.

affects the cost function and the updating value in gradient descent. Hence, the recommendation accuracy with different attenuation parameter α on the accuracy is compared in Figs.9 and 10. If α is larger than 0.025, the ratio of implicit feedback in c_{ui} is large. The average accuracy decreases dramatically. However, if α is too small, we can not differentiate

the effect of implicit feedback on the cost function. In our dataset, it's better to choose the value of α between 0.01 to 0.025. Therefore, α is set 0.025 in the following experiments.

3) TIME

Fig. 11 and 12 show the updating time of different updating methodologies. The y-axis is the log of updating time and the x-axis is the size of data which enter the system. The updating time and accuracy are measured when 2.5% of dataset (about 250,000 records) are analyzed in the system. That is, a unit in the x-axis is 2.5% of dataset, which is called one block.

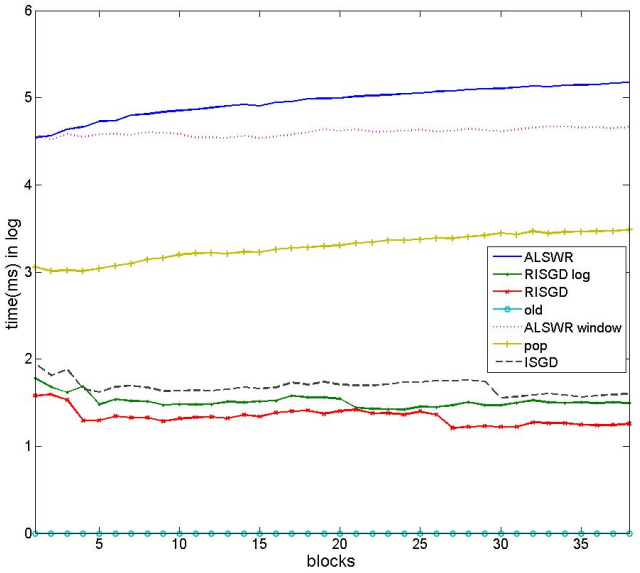

FIGURE 11. The updating time in Case I.

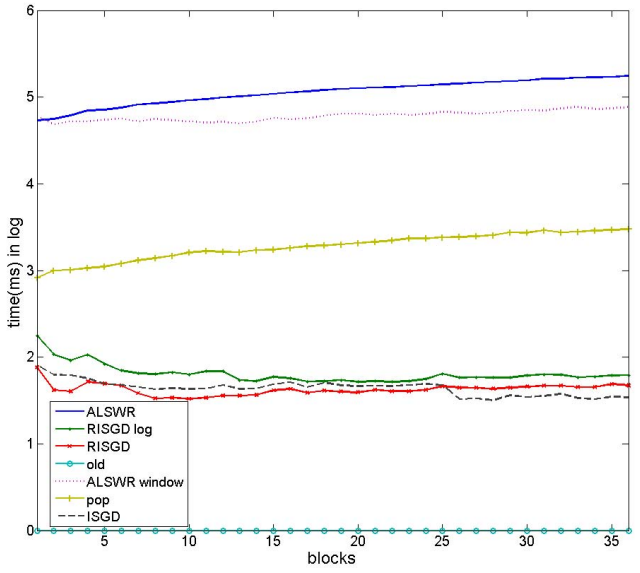

FIGURE 12. The updating time in Case II.

From these figures, the updating time of retraining entire model by ALSWR grows linearly with the size of data. On the other hand, the proposed RI-SGD method only updates the corresponding two latent factors. The updating process of RI-SGD is unrelated to the data size. Similarly, RI-SGD log and ISGD are also unrelated to the data size. About ALSWR window, we set the window size to be 5% and 10% of data. Thus, ALSWR window updates model with 5% and 10% of latest data. As the result, the updating time of ALSWR window by updating with 10% of latest data is twice than the updating time with 5% of latest data, as shown in Table 1.

TABLE 1. Updating time in Case I and Case II.

method	avg. updating time in Case I (ms)	avg. updating time in Case II (ms)
Old model	0	0
RI-SGD	26.85	29.74
RI-SGD log	42.99	47.92
ISGD	48.83	41.37
ALSWR	90170.19	120583.20
ALSWR window	38040.24	75077.52
Pop item method	1926.71	2578.28

On average, the updating time of ALSWR is 90 seconds and 120 seconds in Case I and II respectively. The methods with SGD updating mechanisms, include RI-SGD, RI-SGD log and ISGD consume less than 50 milliseconds to update model in Case I and II. It is about 0.02% of the updating time using ALSWR. Even though the updating time of ALSWR is acceptable in the experiment, the delay becomes longer when more and more users and items in the system. In contrast, the proposed RI-SGD can intensively shorten the updating time no matter how many users and items in real-time applications with streaming implicit feedback.

4) RECOMMENDATION ACCURACY

From Fig. 13 and 14, we can see that recommendation accuracy attenuates with time if there is no updating mechanism applied. The accuracy of the proposed RI-SGD is more stable

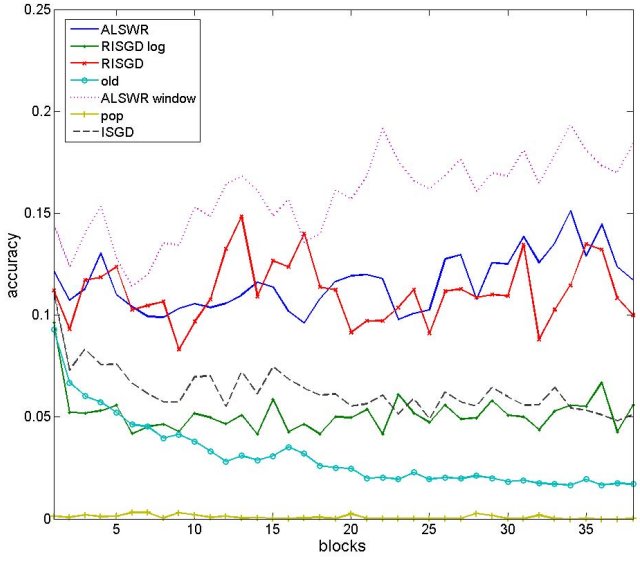

FIGURE 13. Recommendation accuracy in Case I.

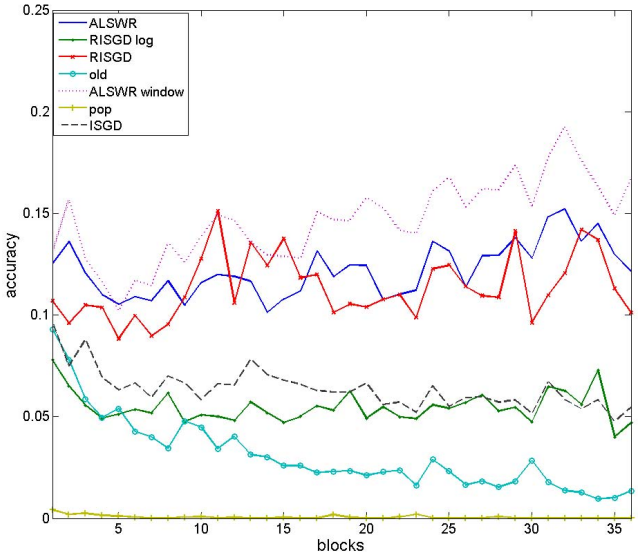

FIGURE 14. Recommendation accuracy in Case II.

and accurate result than ALSWR, which retrains the entire model. That is, the proposed RI-SGD retains accuracy with light-weight update. ALSWR window rebuilds the model based on the latest user behavior to provide the best accuracy among all methods. On the other hand, the accuracy of RI-SGD log, ISGD and recommending popular items are not good. RI-SGD log decreases the accuracy since RI-SGD limits the range of predicted value by eliminating items with large implicit value. Ignoring the numerical value of implicit feedback data makes the accuracy of ISGD be affected a lot. Recommending the most popular items method, which does not consider the preference difference between users, causes lower accuracy.

The recommendation accuracy of different methodologies in Cases I and II are similar. No matter how many training data we use, the updating process of RI-SGD can retain recommendation accuracy.

5) NORMALIZED DISCOUNTED CUMULATIVE GAIN

To evaluate the recommended quality by different updating methodologies, we measure the discounted cumulative gain (DCG). DCG, which is an indicator of ranking quality, measures the usefulness of an item based on its position in the recommendation list [33]. That is, DCG takes the recommendation order into account. In DCG, the recommended items which are more relevant to users are given higher gain. We define recommended items with higher order position in the recommended list to be more relevant items to users in the experiments. The DCG equation is shown below:

$$DCG_k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{log_2(i+1)}, \tag{15}$$

where k represents the number of items that is recommended and rel_i is gain for each recommendation item. In the experiments, when the recommendation item overlaps the real record, we set the gain rel_i to be 1, otherwise 0. Then we divide the DCG value by the number of recommended items N, which is called normalized discounted cumulative gain (NDCG) to show the result in Fig. 15 and 16.

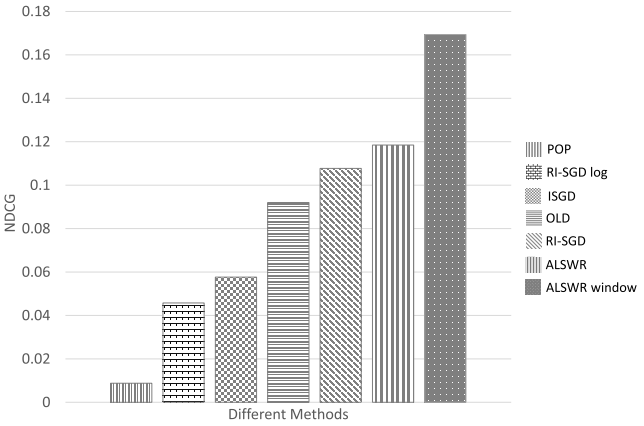

FIGURE 15. The NDCG in Case I.

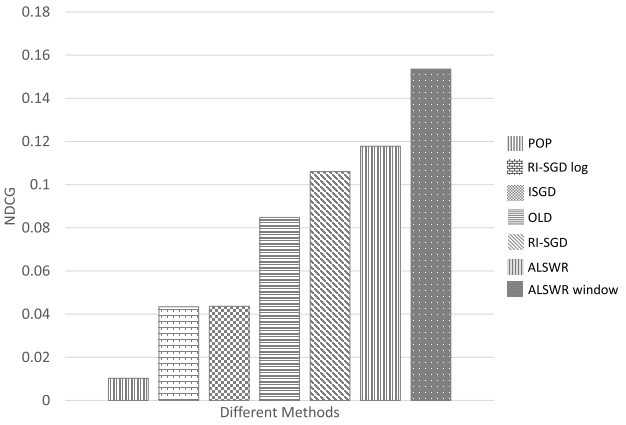

FIGURE 16. The NDCG in Case II.

In the figures, the NDCG of RI-SGD is 90% of NDCG value of ALSWR in Case I and II. That is, RI-SGD can accurately predict the preference degree of users with light-weight updating process.

6) DISCUSSION

We normalize the accuracy and updating time of different updating methods, which are shown in Table 2 and 3. We can

TABLE 2. Normalized accuracy and updating time in Case I.

method	Normalized avg. accuracy	avg. updating time
Old model	0.65	0
RI-SGD	1(10.9%)	1(26.85ms)
RI-SGD log	0.45	1.60
ISGD	0.58	1.82
ALSWR	1.06	3358.29
ALSWR window	1.32	1416.77
Pop item method	0.03	71.76

TABLE 3. Normalized accuracy and updating time in Case II.

method	Normalized avg. accuracy	avg. updating time
Old model	0.62	0
RI-SGD	1(11.1%)	1(29.74ms)
RI-SGD log	0.46	1.61
ISGD	0.56	1.39
ALSWR	1.04	4054.58
ALSWR window	1.42	2524.16
Pop item method	0.04	86.69

see that the accuracy of ALSWR window is the best by adopting the latest data but with long updating time. The accuracy of the proposed RI-SGD is almost same with ALSWR and the updating time of RI-SGD is 1/4055 of ALSWR. All in all, adopting RI-SGD in the recommendation system with continuously streaming implicit feedback data can fast update model and recommend accurate items to users.

VII. CONCLUSION

In this paper, we investigated the accuracy issues of streaming implicit feedback data in recommendation system. We presented a hybrid RI-SGD recommendation systems by combining ALSWR and SGD for the fast updating matrix factorization model with implicit feedback. We defined a time-dependent cost function of matrix factorization and adopted SGD for incrementally updating. Seven different updating methodologies are compared in our experiments. Through experiments and analysis, we show that RI-SGD can achieve almost the same accuracy of ALSWR, but spends only 0.02% of the retraining time of ALSWR. As a result, the real-time and accurate matrix factorization model for streaming implicit feedback can be achieved by differentiating and updating corresponding latent vectors in RI-SGD. In the future, it is worthwhile further extending the proposed incremental updating concept and technique to various collaborative filtering algorithms.

REFERENCES

[1] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2011, pp. 1–35.

[2] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. ACM Int. Conf. World Wide Web*, 2001, pp. 285–295.

[3] *Netflix Prize*. [Online]. Available: https://en.wikipedia.org/wiki/Netflix_Prize

[4] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. ACM SIGKDD Conf. Knowl. Discovery Data Mining*, 2008, pp. 426–434.

[5] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Comput.*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[6] C. Chen, D. Li, Q. Lv, J. Yan, S. M. Chu, and L. Shang, "MPMA: Mixture probabilistic matrix approximation for collaborative filtering," in *Proc. Int. Joint Conf. Artif. Intell.*, 2016, pp. 1382–1388.

[7] G. Zeng, H. Zhu, Q. Liu, P. Luo, E. Chen, and T. Zhang, "Matrix factorization with scale-invariant parameters," in *Proc. Int. Joint Conf. Artif. Intell.*, 2015, pp. 4017–4024.

[8] L. Wu *et al.*, "Modeling the evolution of users' preferences and social links in social networking services," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 6, pp. 1240–1253, Jun. 2017.

[9] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 263–272.

[10] S. Rendle and L. Schmidt-Thieme, "Online-updating regularized kernel matrix factorization models for large-scale recommender systems," in *Proc. ACM Conf. Recommender Syst.*, 2008, pp. 251–258.

[11] K. Wang, H. Peng, Y. Jin, C. Sha, and X. Wang, "Local weighted matrix factorization for top-n recommendation with implicit feedback," *Data Sci. Eng.*, vol. 1, no. 4, pp. 252–264, 2016.

[12] G. Ling, H. Yang, I. King, and M. R. Lyu, "Online learning for collaborative filtering," in *Proc. Int. Joint Conf. Neural Netw.*, 2012, pp. 1–8.

[13] J. Vinagre, A. M. Jorge, and J. Gama, "Fast incremental matrix factorization for recommendation with positive-only feedback," in *Proc. Int. Conf. User Modeling, Adaptation, Pers.*, 2014, pp. 459–470.

[14] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2016, pp. 549–558.

[15] T. Yu, O. J. Mengshoel, A. Jude, E. Feller, J. Forgeat, and N. Radia, "Incremental learning for matrix factorization in recommender systems," in *Proc. IEEE Int. Conf. Big Data*, Dec. 2016, pp. 1056–1063.

[16] S. Funk, "Netflix update: Try this at home," Tech. Rep., 2006. [Online]. Available: http://sifter.org/simon/journal/20061211.html

[17] A. Paterek, "Improving regularized singular value decomposition for collaborative filtering," in *Proc. KDD Cup Workshop*, 2007, pp. 5–8.

[18] C. Ballard, D. M. Farrell, M. Lee, P. D. Stone, S. Thibault, and S. Tucker, *IBM InfoSphere Streams Harnessing Data in Motion*. IBM Redbooks, 2010.

[19] P. Zikopoulos and C. Eaton, *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*. New York, NY, USA: McGraw-Hill, 2011.

[20] *IBM Streams*. Accessed: 2015. [Online]. Available: https://ibmstreams.github.io/

[21] K.-H. Tsai, C.-Y. Lin, L.-C. Wang, and J.-R. Chen, "Reconstruct dynamic systems from large-scale open data," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2015, pp. 1–6.

[22] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[23] J. Bennett, C. Elkan, B. Liu, P. Smyth, and D. Tikk, "KDD cup and workshop 2007," *ACM SIGKDD Explorations Newslett.*, vol. 9, no. 2, pp. 51–52, 2007.

[24] R. M. Bell and Y. Koren, "Scalable collaborative filtering with jointly derived neighborhood interpolation weights," in *Proc. IEEE Int. Conf. Data Mining*, Oct. 2007, pp. 43–52.

[25] P. Matuszyk and M. Spiliopoulou, "Selective forgetting for incremental matrix factorization in recommender systems," in *Discovery Science*. Cham, Switzerland: Springer, 2014, pp. 204–215.

[26] P. Matuszyk, J. Vinagre, M. Spiliopoulou, A. M. Jorge, and J. Gama, "Forgetting methods for incremental matrix factorization in recommender systems," in *Proc. ACM Symp. Appl. Comput.*, 2015, pp. 947–953.

[27] X. Huang, L. Wu, E. Chen, H. Zhu, Q. Liu, and Y. Wang, "Incremental matrix factorization: A linear feature transformation perspective," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 1901–1908.

[28] M. Deshpande and G. Karypis, "Item-based top-n recommendation algorithms," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 143–177, 2004.

[29] *Apache Mahout: Scalable Machine Learning and Data Mining Library*. Accessed: 2014. [Online]. Available: http://mahout.apache.org

[30] Ò. Celma, "Music recommendation and discovery in the long tail," Ph.D. dissertation, Dept. Inf. Commun. Technol., Univ. Pompeu Fabra, Barcelona, Spain, 2008.

[31] *Last.fm Web Services*. Accessed: 2013. [Online]. Available: http://cn.last.fm/api

[32] G. Karypis, "Evaluation of item-based top-n recommendation algorithms," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2001, pp. 247–254.

[33] S. Balakrishnan and S. Chopra, "Collaborative ranking," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2012, pp. 143–152.

CHIA-YU LIN received the B.S. and M.S. degrees in computer science from National Chiao Tung University, Taiwan, in 2010, and 2012, respectively, where she is currently pursuing the Ph.D. degree with the Institute of Communications Engineering. Her current research interests include data-driven real-time updating techniques for recommendation algorithms and mathematical frame-work for data streaming applications.

LI-CHUN WANG (M'96–SM'06–F'11) received the B.S. degree from National Chiao Tung University, Taiwan, in 1986, the M.S. degree from National Taiwan University in 1988, and the M.Sc. and Ph.D. degrees from the Georgia Institute of Technology, Atlanta, in 1995 and 1996, respectively, all in electrical engineering.

From 1990 to 1992, he was with the Telecommunications Laboratories, Ministry of Transportations and Communications, Taiwan (currently the Telecom Labs of Chunghwa Telecom Co.). In 1995, he was affiliated with the Bell Northern Research of Northern Telecom, Inc., Richardson, TX, USA. From 1996 to 2000, he was with the AT&T Laboratories, where he was a Senior Technical Staff Member with the Wireless Communications Research Department. In 2000, he has joined National Chiao Tung University, Taiwan, where he chaired the Department of Electrical and Computer Engineering, where he is currently with the Department of ECE and Department of Computer Science. His recent research interests are focused on the cross-layer optimization and big-data driven methodology for broadband mobile networks, and cognitive communications/computing for latency-critical IoT applications.

He has published over 90 journal papers and 180 conference papers, 18 U.S. patents and co-edited a book Key Technologies for 5G Wireless Systems, Cambridge, in 2016. His recent research interests are focused on the cross-layer optimization and big-data driven methodology, cognitive communications/computing for broadband mobile networks, and industrial IoT. He was elected to the IEEE Fellow for his contributions in cellular architectures and radio resource management in wireless networks. He was a recipient of the 1997 IEEE Jack Neubauer Best Paper Award in 1997, Distinguished Research Award of National Science Council, Taiwan, in 2012, and the IEEE Communications Society Asia-Pacific Board Best Paper Award in 2015.

KUN-HUNG TSAI received the B.S. and M.S. degrees in electrical and computer engineering from National Chiao Tung University, Taiwan, in 2013, and 2015, respectively. His research topic focussed on real-time updating technique for matrix factorization with implicit feedback.