A Distributed Recommender Agent Model based on User’s Perspective SVD Technique

Dussadee Praserttitipong, Peraphon Sophatsathit

Advanced Virtual and Intelligent Computing (AVIC) Center,
Department of Mathematics and Computer Science,
Faculty of Science, Chulalongkorn University, Bangkok, 10330, Thailand.
E-mail: D.Praserttitipong@gmail.com, Peraphon.S@Chula.ac.th

Abstract

This paper proposes a modified centralized recommender technique to be embedded in a software agent. The software agent will act as a cooperative assistant to carry out some tasks on behalf of its user working in the decentralized environment. The historical rating data from other available sources and itself are learned so as to constructing the agent’s prior knowledge at the central server. Knowledge acquisition is accomplished by an eager learning process based on adaptive user’s perspective singular value decomposition (USVD) technique. Subsequent incremental knowledge update according to user’s feedback is maintained by a lazy learning process until some thresholds are reached. Thereby the eager learning process takes over. The overall accumulated knowledge entails the agent to arrive at more accurate recommendations. The processing time for this knowledge maintenance is reduced from polynomial time to constant time. Moreover, the versatility of the proposed model renders it to be applied to sparse knowledge and new-item cold-start problems. This software agent is referred to as a recommender agent.

Keywords: recommender agent, decentralized environment, knowledge acquisition, singular value decomposition.

1. Introduction

An intelligent software agent is a type of software agent having built-in intelligent behavior to handle the tasks using a priori knowledge. In information society, this intelligent agent often serves as a tool for instructing the users to the information they are seeking. Built upon traditional search tools such as search engines, this agent helps their users find relevant service or goods based on the users' profiles or preference [10] in a distributive way. In its most common formulation, this task can be reduced to estimating rating scores of the items under investigation such as services or goods that have not been seen by the users. The results of these systems can take the form of predicted rating scores for some particular items or a set of top-n recommended items.

A number of intelligent agent systems have been presented in literatures [4, 11]. Some of them suggest objects similar to the ones that user preferred in the past. This is referred to as a recommender system. The fundamental of recommendation techniques bring about the outcomes corresponding to user personalized information by learning from their historical decisions. An inference mechanism that corresponds to the user’s needs and contents of the item are initiated to evaluate the reputation score of the unrated items. Some intelligent agent systems construct their decisions by taking advantage of the collaborative knowledge and expertise acquired from other agents. The recommendations are made according to the calculated user’s profiles similarity, where the user’s profiles or preferences are implicitly observed or explicitly requested from the users. Thus, this collaborative-based recommendation system has some key advantages over the content-based techniques, regardless of contents and types of item. However, building comprehensive knowledge of user’s profile from explicit inquiry process adds additional workload that might hinder the advantages of user’s cooperation [4].

In order to model such an intelligent agent system, the collaborative filtering (CF) [1, 2, 3, 5, 7] technique, which is the best known technique in conventional centralized recommender system, is investigated in this paper. Rather than performing profiles matching, the collaboration is established based on similarity values of historical rating scores. Hence, users are free from the burden of providing explicit data that can be obtained elsewhere. The users or agents who have similar tastes
are selected to acquire extra knowledge on unfamiliar items. However, some inherent problems concerning with the centralized CF recommender system still persist. Centralized CF recommender system is constrained by the sparse data, where the number of historical data already obtained is usually very small compared to the number of data needed. This problem affects the accuracy of the recommendation model. This shortcoming is further accentuated by the cold-start problem, where new items cannot be recommended until some users have rated on them. In addition, conventional CF recommender systems generally perform the evaluation at the central repository where user's ratings are stored. This centralized evaluation can induce a bottleneck to the system. A simple solution to distribute the workload of this CF technique is to create knowledge sharing sessions among collaborative agents and retain these connections at all time. However, fully connection across all collaborative agents cannot be guaranteed to succeed because on-line partnership with others might increase the overall network congestion, let alone sessions that might not be successfully established. As a result, some useful information might be lost and the agents have to compute the rating assessment based on incomplete information.

Therefore, further investigations are needed in order to arrive at some form of modified centralized CF technique which are practical enough to be embedded into the agent model working in the decentralized environment. Provision for some limitations of CF technique must be incorporated. Thus, the proposed model will employ an adaptive user's perspective SVD technique, which evolves from the Regularized Singular Values Decomposition (RSVD) technique typically implemented in the centralized CF recommender system. The knowledge from other agents and itself is acquired to construct prior knowledge of the agent. A lazy learning process based on an incremental update concept is also presented to enabling adaptability of the agent. Details will be described as follows. Section 2 presents the proposed model. Section 3 exhibits the experimental results along with discussions. Section 4 concludes with and some final thoughts.

2. Proposed model

A few preliminary terminologies [3, 5] are established for subsequent use as follows:

Let \( r \) be an \( m \times n \) user-item rating matrix. The element \( r_{a,i} \) is a rating score of user \( a \) over item \( i \), and the row \( r_a \) can be represented as \( r_a = (r_{a,1}, r_{a,2}, \ldots, r_{a,n}) \). If item \( i \) has been rated by user \( a \), \( r_{a,i} \geq 1 \).

The SVD-based techniques are inspired by the effective matrix factorization technique from natural language processing [1, 2, 3, 5, 7]. This class of technique is one of the successful CF techniques in conventional centralized recommender system [2]. Several SVD-based techniques such as RSVD [8], improved RSVD model [9], and SVD++ [6] are investigated. The main concept is to factorize matrix \( r \) to smaller matrices relative to variable \( K \). The factorized matrices are set up according to the learning process, where the composition of these matrices must be almost equal to the value of the elements in matrix \( r \).

In the RSVD technique [8], each item is represented by a set of aspects and each user is represented by a set of values indicating their preference for various aspects of the items. The primitive RSVD learning process arrives at the factorized matrices \( p \) and \( q \) by utilizing a minimized squared error function. A baseline estimation for an unknown rating \((\hat{r}_{a,i})\) is assigned as

\[
\hat{r}_{a,i} = \sum_{k=1}^{K} (p_{a,k} \times q_{i,k})
\]

(1)

where \( p_a \) and \( q_i \) are model parameters that their values are specified according to the learning process, variable \( K \) is an appropriate size of factorizing matrices.

The improved RSVD technique [9] extends the prediction accuracy of RSVD technique by adding some bias parameters to RSVD model to indicate the observed bias (deviations) of both user \( a \), denoted by \( b_a \), and the observed bias over the item \( i \), denoted by \( c_i \). The baseline estimation becomes
\[
\hat{r}_{a,i} = \mu + b_a + c_i + \sum_{k=1}^{K} (p_{a,k} \times q_{i,k})
\]  

(2)

where \(\mu\) is the average of all known rating values and \(b_a\), \(c_i\), \(p_a\) and \(q_i\) are model parameters whose value is specified according to the learning process.

Nevertheless, both a primitive RSVD and improved RSVD techniques are constrained by sparse data problems, where the number of known ratings available for learning process is usually very small compared to the number of ratings to be predicted. In addition, an item cannot be recommended unless a user has already rated it. This problem is named as a new item cold-start problem. Thus, the above limitations will be solved by the proposed USVD technique. Three processing steps are proposed, namely, (1) eager learning process, (2) lazy learning process, and (3) predicting process.

### 2.1 Eager Learning Process

The eager learning process is executed at a centralized server (hereafter referred to as R-server), in which the relationships among the agents are created within a finite set of trusted registered agents. The collaboration is established based on evaluation of historical rating scores. The collections of other agents’ attitudes are assembled via an off-line process. Thus, the complete knowledge sharing session across all others collaboration can be assured.

The learning process performs minimization of regularized square error by utilizing a stochastic gradient descent optimization for assessing the prior knowledge to estimate. The baseline estimation for \(\hat{r}_{a,i}\) is given by

\[
\hat{r}_{a,i} = \bar{r}_a + b_a + c_i + \sum_{k=1}^{K} (p_{a,k} \times q_{i,k})
\]  

(3)

where \(\bar{r}_a\) is the average rating score of all known rating values working out by an agent \(a\). The variables \(b_a\), \(c_i\), \(p_{a,k}\) and \(q_{i,k}\) denote learning parameters. This set up permits changing the construction of decision making knowledge model from the overall rating model (i.e. \(\mu\)) to mainly stating a user’s perspective (\(\bar{r}_a\)). This enables more practical knowledge refreshment. Thus, the agent’s adaptation mechanism is in turned loosely based on other agent’s collaboration.

The proposed learning algorithm is described below.

1. Compute an average rating score of all known rating values working out by an agent \(a\), denoted by \(\bar{r}_a\). The value of \(\bar{r}_a\) is determined from

\[
\bar{r}_a = \frac{\sum_{i=1}^{n_a} r_{a,i}}{n_a}
\]  

(4)

where \(n_a\) denotes the number of items that have been rated by user \(a\) and \(n_a \leq n\).

2. Compute the average rating score of all known rating values on item \(i\), denoted by \(\bar{r}_i\). The value of \(\bar{r}_i\) is determined from

\[
\bar{r}_i = \frac{\sum_{a=1}^{m_i} r_{a,i}}{m_i}
\]  

(5)

where \(m_i\) denotes the number of users that have been rated on an item \(i\) and \(m_i \leq m\).

3. Create an \(m_i \times m\) matrix which collects prior knowledge about the user’s variant with respect to each item (\(\beta\)). The elements of matrix \(\beta_{a,i}\) are determined as follows:

3.1 Set all elements in matrix \(\beta\) to 0.

3.2 Randomize the learning parameters matrices \(b_a\), \(c_i\), \(p_a\) and \(q_i\) with small values, where \(b_a\), \(c_i\), \(p_a\) and \(q_i\) are \(m \times 1\), \(n \times 1\), \(m \times k\), and \(n \times k\) matrix, respectively.

3.3 Construct set \(\mathcal{X}\) to collect all rated rating values. The size of this set is represented by variable \(count\) which can be calculated by

\[
count = \sum_{a=1}^{m} n_a
\]  

(6)
3.4 REPEAT

3.4.1 Set $MAE = 0$, $Iter = 0$

3.4.2 FOR any $r_{a,i}$, where $r_{a,i} \in A'$

(a) Calculate the value of $\beta_{a,i}$ as

$$\beta_{a,i} = b_a + c_i + \sum_{k=1}^{K} (p_{a,k} \times q_{i,k})$$

(b) Calculate the predicting rating value $\hat{r}_{a,i}$ as

$$\hat{r}_{a,i} = r_a + \beta_{a,i}$$

(c) Calculate a predicting error as $e_{a,i} = r_{a,i} - \hat{r}_{a,i}$

(d) Evaluate the learning parameters using a stochastic gradient descent technique for minimizing the regularized squared error which informally presented in [6] as follows:

- $b_a \leftarrow b_a + \gamma (e_{a,i} - \lambda \cdot b_a)$
- $c_i \leftarrow c_i + \gamma (e_{a,i} - \lambda \cdot c_i)$
- $q_{i,k} \leftarrow q_{i,k} + \gamma (e_{a,i} \cdot p_{a,k} - \lambda \cdot q_{i,k})$
- $p_{a,k} \leftarrow p_{a,k} + \gamma (e_{a,i} \cdot q_{i,k} - \lambda \cdot p_{a,k})$

(e) Calculate the value of $MAE$, where $MAE = MAE + |r_{a,i} - \hat{r}_{a,i}|$

3.4.3 END FOR

3.4.4 Recalculate the value of $MAE$ using $MAE = MAE / count$

3.4.5 Increment the value of $Iter$ by 1

3.5 UNTIL $Iter = LimitIter$ OR $MAE <= LimitError$

The values of $\gamma$ and $\lambda$ are set to 0.005 and 0.02, respectively [6,12]. The learning process optimization algorithm loops through all ratings in the learning data until the terminal conditions are reached. Normally, there are no explicit terminal conditions for learning process, but in literatures the algorithm loops until the error rate is close to zero or the error rate remains constant. The popular accuracy measurements for recommender system use RMSE (Root Mean Square Error) and MAE (Mean Average Error) [1, 2, 3, 5, 7]. Low RMSE and MAE are resulted from higher accuracy prediction score. The above algorithm sets $MAE$ to be 0.50, whereby the learning process will not lead to overfitting.

According to the learning process, the essential know-how collected from other agents are encapsulated as the prior knowledge. The resulting knowledge from this algorithm comprises of $\bar{r}_i$, $\bar{r}_a$, and $\beta$

(a) The average rating for item $i$ from this learning process ($\bar{r}_i$), is sent back to the information server where the concerning item $i$ is posted.

(b) The remaining information from learning process, which comprises of the average rating score of all known rating values working out by an agent $a$, the prior knowledge about the user’s variant respecting to each item ($\bar{r}_a$ and $\beta_{a,i}$), is sent in response to the agent $a$, when it requests for total renewing its prior knowledge. The value of the prior knowledge of an agent’s decision making variant is given by $\beta_{a,i} = b_a + c_i + \sum_{k=1}^{K} (p_{a,k} \times q_{i,k})$ in case of item $i$ has been rated, otherwise $\beta_{a,i} = 0$.

Finally, all constructive knowledge is consolidated and encapsulated in the agent for subsequent remote execution.

2.2 Lazy Learning Process

When an agent or user $a$ has provided a new explicit rating score on item $i$, the knowledge captured from the learning process must be refreshed in some way. Thus, the collected knowledge at time $t - 1$, is refreshed to be a new knowledge respecting to time $t$. Two incremental adaptation
processes are proposed as the lazy learning processes, namely, incremental adaptation at client site and at I-server site (information server where the item is posted).

At the client site, the value of a new average rating score of all known rating values on item \( i \) at time \( t \) \( (\bar{r}_a^{(t)}) \) is calculated as

\[
\bar{r}_a^{(t)} = y_{t-1}^{(t)} + \theta
\]

where \( y = \frac{n_{a1}^{(t-1)}}{v_{a1}^{(t-1)}} \), \( \theta = \frac{r_{a1}^{(t-1)}}{n_{a1}^{(t-1)}} \) and \( n_{a1}^{(t)} = n_{a1}^{(t-1)} + 1 \). The value of \( \bar{r}_a^{(t)} \) is collected within an agent’s knowledge and it is refreshed by agent itself via an incremental update process at the agent’s site.

At the I-server site, the value of a new average rating score of all known rating values on item \( i \) at time \( t \) \( (\bar{r}_i^{(t)}) \) is calculated as

\[
\bar{r}_i^{(t)} = x_{t-1}^{(t)} + \delta
\]

where \( x = \frac{m_i}{m_i^{(t)}} \), \( \delta = \sum_{t=1}^{T} \frac{r_{i(t-t)}^{(t-t)}}{m_i^{(t)}} \) and \( m_i^{(t)} = m_i + c \). The value of \( \bar{r}_i^{(t)} \) is collected at the information server. It is refreshed via an incremental update process at I-site. This information must be posted to the agent when the new item is launched to an agent or whenever the information about that item is required by an agent.

These processes are performed in real-time which impose a critical time constraint on the update process. Instead of carrying out total agent’s knowledge adaptation via eager learning process which takes approximately \( O(m^3) \), the incremental update process to add one item costs only \( O(1) \).

2.3 Predicting Process

When the new item has been posted at any point in time \( t \), the agent executes its predicting process by means of a baseline rating estimation equation as

\[
\hat{\bar{r}}_{a,i} = \bar{r}_a + \beta_{a,i} + \text{average}(\Theta_a + \Delta_i)
\]

where \( \Theta_a \) and \( \Delta_i \) are temporal variables which denote new learning knowledge of decision making variant with respect to its own point of view and other agents point of view, respectively.

In case of known item, the values of the variables \( \Theta_a \) and \( \Delta_i \) in Equation (3) are given as

\[
\Theta_a = \bar{r}_a^{(t)} - \bar{r}_a \quad \text{and} \quad \Delta_i = \bar{r}_i^{(t)} - \bar{r}_i
\]

Thus, an Equation (3) can be rearranged to

\[
\hat{\bar{r}}_{a,i} = \bar{r}_a + \beta_{a,i} + \frac{\left( (\bar{r}_a^{(t)} - \bar{r}_a) + (\bar{r}_i^{(t)} - \bar{r}_i) \right)}{2}
\]

In case of a new item, the value of the parameters \( \Theta_a \) and \( \Delta_i \) in Equation (3) are given as

\[
\Theta_a = \bar{r}_a^{(t)} - \bar{r}_a \quad \text{and} \quad \Delta_i = \bar{r}_i^{(t)} - \bar{r}_i
\]

since there is no knowledge about \( \bar{r}_i \) when the agent is learning, the deviation can be computed relatively to known \( \bar{r}_a \)

\[
\hat{\bar{r}}_{a,i} = \bar{r}_a + \frac{(\bar{r}_a^{(t)} - \bar{r}_a) + (\bar{r}_i^{(t)} - \bar{r}_a)}{2}
\]

The advantage of this knowledge is for enabling the adaptability to agent knowledge in distributive way which will result in higher prediction accuracy. Moreover, it helps refresh the know-how about new items without invoking eager learning process every time.

Instead of carrying out evaluation at the central repository site, this process takes place at the client’s site in the distributed environment. The fully connection across all collaborative agents is not taken into account. The coupling from others is unfastened because the know-how of others collaborative agents are all encapsulated in its prior-knowledge. Moreover, the cost of processing time for this prediction process is \( O(1) \), which makes its practical enough to perform in the real-time.
3. Results evaluations and discussions

We used data from MovieLens (www.movielen.mun.edu), containing 943 users, 1682 movies, and 100,000 ratings (ranging from 1 to 5). Generally, low MAE value reflects high accuracy of ratings prediction.

There are three aspects to be evaluated and discussed in this section. They are (1) optimal learning values, (2) accuracy evaluation under sparse data, and (3) accuracy evaluation under new item cold-start problem.

3.1 Optimal learning value

The values of two main variables were investigated to arrive at appropriate operating values, namely, size of factorizing matrices ($K$), and accuracy of eager learning process ($L$). Nine learning data sets of different sizes were employed as train and test sets.

3.1.1 Empirically appropriate value of $K$

According to the principle of SVD-based technique, matrix $\mathbf{e}$ must be factorized to smaller matrices relative to variable $K$. To determine an appropriate value of $K$, an experiment was conducted based on RSVD [13] and improved RSVD [11, 12]. The principal independent variable was $K$ using three trial values, i.e., 100, 300, and 400. In addition, the terminal conditions were designated on two more variables, i.e., MAE and RMSE. The former controlled the learning process until reaching MAE ≤ 0.1. The latter utilized the same threshold, that is, RMSE ≤ 0.1. Table 1 shows the results of this experiment.

<table>
<thead>
<tr>
<th>% of learning data compared with overall data</th>
<th>$K$=100</th>
<th>$K$=300</th>
<th>$K$=400</th>
<th>$K$=100</th>
<th>$K$=300</th>
<th>$K$=400</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td>RMSE</td>
<td>MAE</td>
<td>RMSE</td>
<td>MAE</td>
<td>RMSE</td>
</tr>
<tr>
<td>10%</td>
<td>0.9284</td>
<td>4.9981</td>
<td>0.9190</td>
<td>4.9990</td>
<td>0.7856</td>
<td>0.7554</td>
</tr>
<tr>
<td>20%</td>
<td>0.8140</td>
<td>4.9996</td>
<td>0.8051</td>
<td>4.9998</td>
<td>0.7598</td>
<td>0.7342</td>
</tr>
<tr>
<td>30%</td>
<td>0.7794</td>
<td>4.9995</td>
<td>0.7673</td>
<td>0.7693</td>
<td>0.7457</td>
<td>0.7264</td>
</tr>
<tr>
<td>40%</td>
<td>0.7602</td>
<td>0.7602</td>
<td>0.7468</td>
<td>0.7589</td>
<td>0.7381</td>
<td>0.7169</td>
</tr>
<tr>
<td>50%</td>
<td>0.7454</td>
<td>0.7464</td>
<td>0.7294</td>
<td>0.7445</td>
<td>0.7262</td>
<td>0.7144</td>
</tr>
<tr>
<td>60%</td>
<td>0.7365</td>
<td>0.7410</td>
<td>0.7210</td>
<td>0.7394</td>
<td>0.7232</td>
<td>0.7112</td>
</tr>
<tr>
<td>70%</td>
<td>0.7233</td>
<td>0.7344</td>
<td>0.7099</td>
<td>0.7338</td>
<td>0.7158</td>
<td>0.7038</td>
</tr>
<tr>
<td>80%</td>
<td>0.7235</td>
<td>0.7345</td>
<td>0.7096</td>
<td>0.7155</td>
<td>0.7106</td>
<td>0.7035</td>
</tr>
<tr>
<td>90%</td>
<td>0.7065</td>
<td>0.7191</td>
<td>0.6916</td>
<td>0.6997</td>
<td>0.6889</td>
<td>0.7008</td>
</tr>
</tbody>
</table>

As can be seen in Table 1, bold figures represent the least MAE results for each set while underlined figures reflect unsatisfied test results. The test accuracy of RSVD model given $RMSE < 0.1$ at 10%-20% of learning data yields a significant difference from others. This unsatisfied test accuracy was obtained with high MAE. We found that there were a few differences between test accuracy setting $K$ to be 100, 300, and 400. It turned out that $K=400$ was better than others values. There were also minor test accuracy fluctuations caused by terminating conditions using $MAE \leq 0.1$ or $RMSE \leq 0.1$. However, the results obtained from fixed terminating conditions with $MAE \leq 0.1$ were better than those from $RMSE \leq 0.1$. Thus, it can be concluded that the appropriate suggested value of $K$ is 400 and the terminating condition of RSVD and improved RSVD learning process could set $MAE \leq 0.1$. 

Table 1: Prediction accuracy on different values of $K$
3.1.2 Empirically appropriate accuracy value of $\hat{r}$

To evaluate an appropriate accuracy value for terminating condition of eager learning process of USVD, the experiment was conducted based on a non-adaptive USVD (without lazy learning process) setting the value of $\hat{r}$ to be 400. The terminating condition was designated on $MAE$. The learning process was repeated until $MAE \leq L$, where $L$ varied among $0.1, 0.3, 0.5,$ and $0.7$.

Table 2: Prediction accuracy on different values of $\hat{r}$.

<table>
<thead>
<tr>
<th>% of learning data compared with overall data</th>
<th>A non-adaptive USVD fitting learning accuracy with $MAE \leq L$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$L = 0.1$</td>
</tr>
<tr>
<td>10%</td>
<td>0.8024</td>
</tr>
<tr>
<td>20%</td>
<td>0.7486</td>
</tr>
<tr>
<td>30%</td>
<td>0.7335</td>
</tr>
<tr>
<td>40%</td>
<td>0.7180</td>
</tr>
<tr>
<td>50%</td>
<td>0.7073</td>
</tr>
<tr>
<td>60%</td>
<td>0.7056</td>
</tr>
<tr>
<td>70%</td>
<td>0.6949</td>
</tr>
<tr>
<td>80%</td>
<td>0.6888</td>
</tr>
<tr>
<td>90%</td>
<td>0.6758</td>
</tr>
</tbody>
</table>

As shown in Table 2, test accuracy from a non-adaptive USVD with fitted learning accuracy at $MAE \leq 0.5$ yielded good scores in every case. Hence, the appropriate accuracy value for terminating condition of the proposed USVD eager learning process is 0.5.

The effectiveness of the lazy learning process was computed based on an adaptive USVD. The accuracy value of terminating condition of the learning process was initially set at $MAE \leq 0.5$. The feedback rating values from the user agent assumed to be sent back after each predicting result was posted. In addition, the incremental update process at both agent and I-Server were performed after each feedback data.

![Figure 1: Comparison of MAE returned from RSVD-based techniques on different percent of learning data](image-url)
Figure 1 shows test accuracy results achieved from an adaptive USVD (solid-line) surpassing both RSVD and improved RSVD techniques in every aspect. This is because the adaptive USVD incorporates the lazy learning process as its main task. Thus, the agent’s knowledge is always up-to-date, boosting the accuracy of the agent’s decision making. Note that the non-adaptive USVD technique also provides good test accuracy results where the number of learning data exceeds 20%. This implies that under distributed environment and off-line situations, no adaptation (lazy learning) process should be performed. The agent could implement its decision making process by relying on the original knowledge from the USVD eager learning process.

3.2 Accuracy evaluation under sparse data problem

Since 97.3% of data set encompassing 943 users, 1682 movies, an 100,000 rating was mostly sparse, we resorted to a standard 5-fold cross-validation based on 80-20 off-line learning and test sets, respectively. We compared our technique with a number of existing techniques, namely, Naive Bayes classification approach (NB) [15], SVM classification technique [13], collaborative filtering via co-clustering(COCL) [14], evolutionary co-clustering (ECOCL) [12], LSI/SVD [10], RSVD [8], improved RSVD [9], and SVD++ [6]. Only NB and SVM are content-based recommendation techniques, the rest are CF recommendation techniques. The results of accuracy comparisons are shown in Figure 2.

![Figure 2: Comparison of prediction accuracy under sparse data problem](image)

As can be seen from Figure 2, the MAE of test accuracy results from the proposed adaptive USVD technique is the lowest. This is due to its adaptation ability. The non-adaptive USVD technique also yields the comparable results. Thus, the proposed USVD is better than other conventional CF techniques spare data problem.

3.3 Accuracy evaluation under new item cold-start problem

To determine the accuracy of the proposed approach under the new item cold-start problem, the experiment was conducted based on different number of new test items. The size of the new items varied from 10%-80% in comparison with that of the overall datasets. Some parameters had to be adjusted to compensate for the shortcoming of CF technique under the new item cold-start problem. In this case, we employed a prediction process based solely on new added parameters, namely, $\bar{r}_i^{(t)}$, $\bar{r}_j^{(t)}$, $\text{average}(\bar{r}_i^{(t)} + \bar{r}_j^{(t)})$ and $\text{average}(\Theta_u + \Delta_i)$. Different sizes and parameters of new test items were used, ranging from 10-80% of the size of overall data sets. The accuracy results are shown in Table 3.
Table 3. Prediction accuracy evaluation under new item cold-start problem

<table>
<thead>
<tr>
<th>% of new items compared with learning datasets</th>
<th>% of learning data compared with overall data</th>
<th>Decision made on</th>
<th>( \bar{r}_a )</th>
<th>( \bar{r}_a^{(t)} )</th>
<th>( \bar{r}_i^{(t)} )</th>
<th>average((\bar{r}_i^{(t)} + \bar{r}_a^{(t)}))</th>
<th>( \bar{r}_a + \text{average}(\Theta_a + \Delta_i) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(10/90) 11%</td>
<td>90%</td>
<td></td>
<td>0.9117</td>
<td>0.7854</td>
<td>0.8210</td>
<td>0.7699</td>
<td>0.7574</td>
</tr>
<tr>
<td>(20/80) 25%</td>
<td>80%</td>
<td></td>
<td>0.8630</td>
<td>0.7596</td>
<td>0.8059</td>
<td>0.7496</td>
<td>0.7317</td>
</tr>
<tr>
<td>(30/70) 43%</td>
<td>70%</td>
<td></td>
<td>0.8681</td>
<td>0.7580</td>
<td>0.7950</td>
<td>0.7489</td>
<td>0.7326</td>
</tr>
<tr>
<td>(40/60) 67%</td>
<td>60%</td>
<td></td>
<td>0.8604</td>
<td>0.7534</td>
<td>0.7973</td>
<td>0.7433</td>
<td>0.7296</td>
</tr>
<tr>
<td>(50/50) 100%</td>
<td>50%</td>
<td></td>
<td>0.8552</td>
<td>0.7508</td>
<td>0.8017</td>
<td>0.7410</td>
<td>0.7257</td>
</tr>
<tr>
<td>(60/40) 150%</td>
<td>40%</td>
<td></td>
<td>0.8647</td>
<td>0.7541</td>
<td>0.8024</td>
<td>0.7447</td>
<td>0.7292</td>
</tr>
<tr>
<td>(30/70) 233%</td>
<td>30%</td>
<td></td>
<td>0.8694</td>
<td>0.7617</td>
<td>0.8035</td>
<td>0.7502</td>
<td>0.7348</td>
</tr>
<tr>
<td>(80/20) 400%</td>
<td>20%</td>
<td></td>
<td>0.8707</td>
<td>0.7605</td>
<td>0.8051</td>
<td>0.7510</td>
<td>0.7389</td>
</tr>
</tbody>
</table>

It was apparent that eager learning process alone yielded the worst score in cold-start situation where adding a simple parameter such as \( \tau_a \) would result in high MAE values. As lazy learning was supplemented \( \bar{r}_a^{(t)} \), the scores improved, i.e., average\((\bar{r}_i^{(t)} + \bar{r}_a^{(t)})\) (from Eq. 9 & 10) and \( \bar{r}_a + \text{average}(\Theta_a + \Delta_i) \) (from Eq. 13). The improved test accuracy results precipitated directly from this prompt refreshment of the value of the decision parameters. As for the initial stage of evaluation on a new item in a conventional recommender system, there was no user to rate on the item. Thus, the agent had no knowledge to make a decision. This is not the case for the proposed USVD technique which can eventually overcome the new item cold-start problem.

Although CF recommendation techniques, such as NB and SVM, can perform their evaluations with the inclusion of new items, they are not effective for analyzing the ratings of new items having no characterizing profiles. On the other hand, the proposed USVD technique can perform on any kind of items without the need for accompanying characteristics to be used in the prediction process. In addition, the result returned from the proposed USVD technique at the 80% of training data (shown in Table 3. With MAE =0.7317) was better than both NB (depicted in Figure 2 with MAE =0.8125) and SVM (depicted in Figure 2 with MAE =0.7750). It can be concluded that the proposed approach can overcome the drawback of CF technique under the new item cold-start problem.

4. Conclusion

This paper proposes a distributed recommender agent model based on CF technique. The knowledge from other agents and itself are learned to construct prior knowledge residing at the central repository site via an eager learning process. A prediction process can be carried out at the agent site in a distributive way. The lazy learning process incrementally updates on this prior knowledge adaptability ensemble. Fast incremental update processes of prior knowledge are devised to enhance the adaptability and performance of the agent in \( O(1) \). The experimental results confirm the contributions to both sparse knowledge and new item cold-start problem. Since the proposed model is restricted to discrete rating scores, further investigation on continuous rating score will be investigated. As such, it will in turn enable a more general prediction process that is applicable to wider range of applications and will remain to be a challenging research endeavor to be explored.

5. References


