Collaborative Filtering by Using a Graph Partitioning Method on Binary Data

Kansai University, Takanobu NAKAHARA*
Osaka Prefecture University, Hiroyuki MORITA**

Abstract: Collaborative filtering methods are frequently used for predicting users’ preferences in recommender systems, such as those used for recommending movies, music, or articles. These methods have a large impact on businesses, because the volume of content they offer is tremendous, and it is important to support users’ ability to make informed choices from among the available content. To increase sales and improve customer loyalty, many e-commerce companies, such as Amazon and Netflix, have adopted recommender systems. However, these companies generally rely on user ratings for the content they offer, and it is usually difficult or expensive to obtain such ratings data. Hence, we need a high-quality recommender system that uses only binary data, such as historical purchasing data, without ratings. Binary data, however, is very simple, and it is therefore difficult to express a relationship in detail between users and items by using only such simple methods. This paper proposes a recommender system based on a graph-partitioning method to solve the problems through a two-phase approach: We generate a model that expresses the relationships between various items and implements an appropriate grouping by using a graph-partitioning method. We then use our proposed algorithms to determine accurate recommendations. A comparison of our results with those obtained from traditional methods reveals that our method is more practical for businesses usage.

Keywords: Recommender system, Graph partitioning, Collaborative filtering, Binary Data, Data mining.

INTRODUCTION

Recommender systems are becoming increasingly important in e-business, as people access e-commerce Web sites with greater frequency. In the past decade, many studies, such as those by Hofmann (1999), Pennock et al. (2000), Herlocker et al. (1999), Resnick et al. (1994), Sarwar et al. (2001), Melville et al. (2002), and Rashid et al. (2006) have proposed recommendation algorithms for these sites. As Melville et al. stated, these studies have been categorized into two main fields: collaborative filtering (CF) and content-based recommendation (CB). The former approach is advantageous because it does not require associating descriptive content with items but needs only ratings data, and it can also recommend untested or unproven
items in terms of what similar users have recommend-
ed. However, there are also two fundamental weak points. One is that very large amounts of data create a very sparse user-item ratings matrix, which means few similar users. The other is the so-called first-rater problem, which states that it is difficult to apply CF algorithms to unrated items.

In addition to these points, we must call attention to a few problems from the viewpoint of practical usage. First, the CF algorithms need an abundance of ratings to recommend items, so firms have to collect ratings data. They easily accumulate point-of-sale (POS) data or panel data, but it is difficult to collect ratings data because of the additional costs needed. We can apply certain binary, such as POS data, to existing CF algorithms, but the performance is usually poor. From the problems we have thus far encountered, we can determine that we need to respond to large, sparse data and that it is important to increase the accuracy of recommendations for binary data such as POS data.

We propose an algorithm that uses the relationship among purchased items, although additional ratings data can also be used. The algorithm consists of two phases: clustering items using the graph-partitioning method and selecting items for recommendation from the clusters. In the selecting phase, we propose a selecting strategy using the D’Hondt method in order to reflect users’ preferences. Using computational experiments gathered from historical purchasing data of CDs, we illustrate that the proposed algorithm performs well.

The remainder of this paper is organized as follows. Section 2 reviews related works. Section 3 proposes the new recommender system using a graph-partitioning method. Section 4 provides a brief overview of the traditional user-based and item-based CF approaches. Finally, Section 5 discusses criteria and reports on the experimental results.

RELATED WORKS

The CF algorithm is the most popular approach for recommender systems and can be broadly broken into two classifications: user-based type and item-based type. The former is applied to find users with similar characteristics, the latter to find similar items from a data set. Although the viewpoints in the two approaches are different, the computational complexities are the same, as represented by O(mn), where m and n are the number of items and users, respectively. Of course, the magnitudes of m and n determine whether O(mn) is large or small. Generally, m and n are large, and consequently, these algorithms require considerable time to generate data for large online data sets. As a result, the algorithm can be used as an offline system, but it is difficult for use with an online system. Using the algorithm on a small data set could overcome this problem, but doing so might lead to poor recommendations and is therefore impractical. As an alternative, Rashid et al. (2006) propose a two-step method by using k-means clustering and the CF algorithm, in a process called ClustKNN.

In the first step of the ClustKNN method, users with similar characteristics are divided into k groups using k-means clustering. Then the CF algorithm makes a recommendation using k surrogate users. Although the first offline step has a time complexity of O(mn), the second online step has a time complexity of only O(m), because k is sufficiently small. The division of the search process into offline and online modes is an efficient method of cost reduction, since the offline cost will not affect the user. However, although ClustKNN is a compelling method of solving large sparse data, it usually uses ratings data as the input.

Demiriz (2004) proposes an algorithm using binary data. In this method, association rules between items are discovered as the first step and similarity is then computed following the association rules. This, too,
is an attractive approach, because it does not use the
correlation coefficient as its customary CF algorithms.
But it is expensive to locate huge association rules, and
the rules are unstable for parameters, such as support
value and confidence.

In this paper, we propose a new recommendation
algorithm incorporating ideas from the papers cited
above. Our algorithm is similar to ClustKNN in that
both are two-step algorithms and use a clustering tech-
nique. In our method, we use the support value to de-
fine a similarity between items and divide them into k
similar-item groups. Note that ClustKNN divides users
into k groups by k-means clustering. To reduce calcula-
tion costs, we create some meaningful subitem sets
by using a graph-partitioning technique. From these
subsets, we can use the D'Hondt method as a selecting
strategy to reflect users' preferences, whereas Clust-
CNN uses the general CF algorithm using k surrogate
users. In these processes, we need computational time
using graph-partitioning at offline, and as a second
step, searching costs at online are determined by using
O(m). It is expected that our method can make precise
recommendations using only binary data.

PROPOSED METHOD

In this section, we describe the proposed recom-
mender system using a graph-partitioning method
to apply to binary data.

Given a user u(u=1, ..., n), an item i(i=1, ..., m),
and each user's preferences on items \( R_{ui} = \{0, 1\} \),
we suppose that 1 and 0 denote purchase and no pur-
chase, respectively. The system performs two types of
computational tasks: the offline task of clustering by
using a graph-partitioning method and the online task
of generating predictions and recommendations. In
the offline task, the purchasing data are transformed
into graph-structured data to express the relationships
between the items. Then the data is partitioned into k
subgraphs by using a graph-partitioning method. In the
online task, the recommended items are determined
using a proposed selecting strategy, as we describe
later. The difference between graph-partitioning and a
general clustering method such as the k-means method
is in whether the balance of the subgraph (or cluster)
is retained or not. A graph-partitioning method can
divide into k subgraphs that have similar sizes for any
data, but general clustering methods sometimes create
grouping imbalances.

Application of graph-partitioning

Graph-partitioning is a NP-hard problem that di-
vides a graph into several subgraphs, keeping a balance
of the vertical size with minimum cutting costs (Garey
1974). The graph-partitioning approach is widely ap-
plied to several fields, including network design, VLSI
design, and data mining.

In our method, we select recommend items from
balanced item graphs. So, to apply graph-partitioning,
we transform the POS data into a weighted undirected
graph. In a graph \( G(V,E) \), \( V \) is a set of vertices and \( E \)
is a set of edges connecting two vertices in \( V \). Given
a positive integer \( k(k \leq m) \), \( V \) is partitioned into \( k \)
subgraphs—that is, \( V = V_1 + V_2 + \ldots + V_k \) (disjoint union)—
which keep a balance with regard to the vertical size,
as shown in equation (1):

\[
w(V_q) \leq p \cdot \frac{w(V)}{k},
\]

where the value of \( p \) is obtained as 1.03 from a
parameter of METIS\(^3\) (Karypis 1998), and, \( w(V_q) \) and
\( w(V) \) denote the sum of weight of the vertices in \( V_q \)
\((q=1, ..., k)\) and \( V \) respectively. The primary goal is to
minimize the sum of the cost of the edges between the
subgraphs, whose cost is expressed as shown in equa-
tion (2):

\[
\sum_{(x,y) \in E, x \in V_j, y \in V_j, i \neq j} c((x,y))
\]

NII-Electronic Library Service
where \( c((x,y)) \) denotes the cost of edge between \( x \) and \( y \). We have to determine suitable values for the costs and weights. The costs of the edges by the strength of the relationships between the items are desired. Here, we use "support" value (Tan 2002), as below:

\[
c((x,y)) = \frac{|U_{x,y}|}{|U|},
\]

(3)

where \( U \) denotes a set of all users, \( U_{x,y} \) denotes a set of users who purchase item \( x \) and \( y \), and \(|\cdot|\) denotes a number of elements in a set. About the weight of the vertex, we give a unit weight, because it is desirable that the number of items is identical in our method.

By applying the graph-partitioning method, the obtained subgraphs exhibit notable properties, one of which is that the total weight of the vertices is almost the same at each subgraph because of the balance of the vertical size. In other words, any subgraph has almost the same number of items. Therefore, this property shows that we can recommend items in the same manner even if we select a subgraph randomly. Another property is that the retained edges in the subgraph show strong relationships as compared to eliminated edges. We decide on recommend items from each subgraph independently, so it is expected that calculation costs can be reduced.

**Recommendation strategies**

Typically, the recommendation is a ranked list of items that the target user will like. This type of recommendation is also known as the Top-N recommendation (Herlocker 2004), which is generated by sorting the results of the prediction rating for all the items not yet rated. In our recommender system, to generate the Top-N list, we sort the total of the edge cost, which denotes the total number of users who rate both items.

We illustrate the manner of selecting the items, which consists of a Top-N list, from each subgraph.

Figure 1 shows the subgraphs obtained from the graph-partitioning method. The bold circles represent the purchased items, \( V_{pu} = \{1,3,6,8\} \). The edges between \( V_{pu} \) are the purchased items and non-purchased items, \( V_{npu} = \{2,4,5,7,9,10\} \) are remained, and the other edges (11,2), (3,6), and (5,12) are eliminated. Finally, the value of the recommendation is calculated as the sum of the cost of the edges that connect each \( V_{npu} \); for example, the value of the recommendation of 4 is 0.6, because 0.2 and 0.4 are connected from vertex 1 and 8, respectively, and are added. The predicted value of each \( V_{npu} \) is calculated in this manner.

![Figure 1: Generating the Top-N list](image)

When we generate the Top-N list, we adopt two types of strategies, as follows.

- **Strategy 1**: The Top-N list is generated by sorting the value of the recommendations on all candidate items.
- **Strategy 2**: The Top-N list is generated using the D'Hondt method, which has been used for allocating seats in proportional-representation systems.
Strategy 1 emphasizes that the recommended items have a greater relationship with the purchased items. If the value is high, it shows that the item is connected strongly with purchased items. However, this strategy selects connected items too strong for recommending more diverse items. On the other hand, Strategy 2 decides the number of the recommended items from each subgraph in the proportion of the number of purchased items in $V_q$ with the relative size—that is, the number of purchased items in each subgraph to all purchased numbers in $V$. In this case, a subgraph is $q (q = 1, \ldots, k)$ and $H_{iq}$ is the number of items in $V_q$ purchased by a target user $t$. $N$ denotes the number of recommended items. And $C_{iq}$ denotes the number of recommended items for each $V_q$. The algorithm of the D’Hondt method includes the following.

Step 1: Initialization of $C_{iq} = 0$
Step 2: Select $q$ which is maximum $H_{iq}$
Step 3: $C_{iq}++$
Step 4: $H_{iq}$ is updated by $H_{iq} / (C_{iq} + 1)$
Step 5: While $\sum_q C_{iq} \leq N$, repeat from step 2 to 4

Finally, the number of recommended items is decided from each subgraph by using each Top-$N$ list. Later, we compare the two types of strategies.

**EXISTING COLLABORATIVE FILTERING**

To compare our proposed method with traditional CF systems, we describe two types of CF systems: user-based CF and item-based CF. We also describe a method that involves the use of only random sampling and is the simplest available technique. ClustKNN is similar to our proposed methods. However, we compare our proposed method to the traditional CF systems since the user-based CF has better performance than ClustKNN does (Rashid, 2006). We shall now provide a brief overview of each algorithm.

**User-based CF algorithm**

In this algorithm, the rating prediction is obtained from the similarity of user preferences. The basic idea is that users with similar characteristics will have similar preferences. This algorithm consists of a two-step process. First, the similarities between users are computed by common items that have been purchased by both the target user and all other users. Although several techniques are available to determine similarities, we selected the Pearson correlation, which is the most commonly used technique (Resnick, 1994). The similarities of the users are computed by using equation (4):

$$simU(u, tu) = \frac{\sum_{i \in I} (R_{ui} - \overline{R}_u)(R_{ti} - \overline{R}_t)}{\sqrt{\sum_{i \in I} (R_{ui} - \overline{R}_u)^2} \sqrt{\sum_{i \in I} (R_{ti} - \overline{R}_t)^2}}$$

(4)

where $tu$ denotes the target user, and $I$ is the set of items purchased by both $u$ and $tu$. $\overline{R}_i$ is the average of user $u$ on all the items.

Next, the rating prediction $\hat{R}_{ui,tu}$ is computed by using the similarity among users, obtained from step one, and performing a weighted average of the deviations from the selected user means:

$$\hat{R}_{ui,tu} = \overline{R}_{tu} + \frac{\sum_{i \in UC} (R_{ui} - \overline{R}_u) \cdot simU(u, tu)}{\sum_{i \in UC} simU(u, tu)}$$

(5)

where $ti$ denotes the target item, and $UC$ is the set of users who are $simU(u, tu) \geq a$. Here an appropriate value is given to $a$ from 0 to 1. The Top-$N$ list is generated from sorting the result of the prediction rating.

**Item-based CF algorithm**

This algorithm also consists of a two-step process. It first computes the similarity between the items and then obtains a prediction by using this similarity. One fundamental difference between the similarity computation in the user-based CF and the item-based CF...
is that in the former, the similarity is obtained from common items purchased by two different users and, in the latter, from users who have purchased two different items. In other words, the difference is either computed along the rows or the columns of the matrix. The similarity between two items $i$ and $b$ is computed by adjusting the cosine measure in equation (6), which was proposed by Sarwar (2001):

$$sim_I(i,b) = \frac{\sum_{u\in U} (R_{u,i} - \bar{R}_u)(R_{u,b} - \bar{R}_u)}{\sqrt{\sum_{u\in U} (R_{u,i} - \bar{R}_u)^2 \sum_{u\in U} (R_{u,b} - \bar{R}_u)^2}},$$

where $U$ denotes the set of users who have purchased both $i$ and $b$.

Once the similarities between the items are computed, the rating space of the target user $u$ is examined to find all the purchased items similar to the target item $ti$. The rating prediction $\hat{R}_{ni,i}$ is computed using the weighted average by equation (7):

$$\hat{R}_{ni,i} = \frac{\sum_{b \in N} \left(sim_I(ti,b) \cdot R_{u,b}\right)}{\sum_{b \in N} \left|sim_I(ti,b)\right|},$$

where $N$ denotes the set of items that are $sim_I(ti,b)$ $\geq \beta$. As similar with $a$, an appropriate value is given to $\beta$ from 0 to 1. Typically, the threshold of $\beta$ similar items is used rather than that of all the items. The Top-N list is generated by sorting in the same manner as the item-based CF.

**Methods of random sampling**

For comparison, a random sampling method for the Top-N list is implemented. In experiments, this method is typically considered to be the standard of the lowest performance.

**EXPERIMENTAL EVALUATION**

**Evaluation metrics**

To evaluate the quality of the recommendations, the F1-measure, which is widely used in recommender systems and calculated on a basis of precision and recall, is used. In the case of ratings data, there are two types of ideas for calculating them. Figure 2 denotes a set of test data. In the figure, $A$ and $B$ denote an item set existing in test data with a rating of 1, 2, or 3 and in test data with a rating of 4 or 5, respectively. $D$ denotes an item set existing in a training data set. And $C$ denotes the other item set in $I$, that is $I = A \cup B \cup C \cup D$. Two types of ideas are proposed with this set.

<table>
<thead>
<tr>
<th>Training data set (D)</th>
<th>Test data set with rating of 4 or 5 (B)</th>
<th>Test data set with rating of 1, 2, or 3 (A)</th>
</tr>
</thead>
</table>

Figure 2: Data sets

One idea calculates precision and recall using $A$, $B$, and $SX$, where $SX$ denotes the Top-N list in $A \cup B$. Generally, the top-N list is generated by sorting the results of the prediction rating for all unrated items. Then, the precision and recall are defined as below.

$$precision = \frac{|SX \cap B|}{|SX|}. \quad (8)$$

$$recall = \frac{|SX \cap B|}{|B|}. \quad (9)$$

Another idea calculates precision and recall using $A$, $B$, $C$, and $SY$, where $SY$ denotes the Top-N list in $A \cup B \cup C$. Then, the precision and recall are defined as below.
\[
\text{precision} = \frac{|SY \cap B|}{|SY|}, \quad (10)
\]
\[
\text{recall} = \frac{|SY \cap B|}{|B|}, \quad (11)
\]

In this idea, although the values of both criteria tend to be small, the criteria are more practical, because recommended items are selected from all candidate items. Almost all existing research used the former idea to evaluate the quality of recommendations in ratings data. However, we cannot use those former ideas, since we are using binary data. In our case, \( B \) consists of items with a rating of 1, and \( A \) is empty. Therefore, \( SY \) denotes the Top-N list in \( B \cup C \). So we modify the equation (10) and (11), and calculate precision and recall. Then the F1-measure is calculated as below.

\[
F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \quad (12)
\]

Data set

To evaluate these methods, we used historical purchasing data of CDs, sourced from a certain CD store in Japan. The collected data covers 1,200,000 records, 330,000 customers, and 50,000 items over a two-year period. Figure 3 shows the image of data sets. We divided the purchasing data into the first and second years to be used for training sets and test sets, respectively. We selected 1,340 users who purchased more than 15 CDs in the first year and more than five CDs in the second year. We determined target users, who consisted of 10% of the 1,340 users, by using a tenfold cross validation. Therefore, the recommended items for the target users are determined by using training sets.

Results and discussion

We compare the proposed method with the other methods reviewed in the previous section by using the above-mentioned evaluation metrics. For the experiments, we set parameters for the graph-partitioning and item-based CF. In the method using graph-partitioning, the number of partitions is \( k = 30 \). In the case of the user-based CF and item-based CF, we used \( a = 0.1 \) and \( \beta = 0.2 \), respectively. Better values are selected from our preliminary experiments.

Tables 1 and 2 show Top-N recommendation results with \( N = 5 \) and 10, respectively, using precision, recall, and F1-measure. The Top-N list generated by Strategy 2 of our proposed method, that is, “graph (Strategy 2),” produced the best quality of all the metrics in both the Top-5 and Top-10 recommendation results. Next, the second-best quality was obtained by the Strategy 1, that is, “graph (Strategy 1).” A comparison of our method with the traditional CF methods reveals that the user-based CF delivers a better performance than the item-based CF. Although the values of all the metrics are poor on the whole, they are better than random sampling. Therefore, we can show that it is very difficult to select an item that we can say with certainty is going to be purchased in the near future.

<table>
<thead>
<tr>
<th>Item</th>
<th>precision</th>
<th>recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>graph (Strategy 2)</td>
<td>0.0121</td>
<td>0.0078</td>
<td>0.0094</td>
</tr>
<tr>
<td>graph (Strategy 1)</td>
<td>0.0087</td>
<td>0.0052</td>
<td>0.0065</td>
</tr>
<tr>
<td>user-based</td>
<td>0.0082</td>
<td>0.0051</td>
<td>0.0062</td>
</tr>
<tr>
<td>item-based</td>
<td>0.0016</td>
<td>0.0013</td>
<td>0.0014</td>
</tr>
<tr>
<td>random sampling</td>
<td>0.0009</td>
<td>0.0005</td>
<td>0.0006</td>
</tr>
</tbody>
</table>
Table 2: Results of the Top-10 recommendation using precision, recall, and F1-measure

<table>
<thead>
<tr>
<th></th>
<th>precision</th>
<th>recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>graph (Strategy 2)</td>
<td>0.0099</td>
<td>0.0131</td>
<td>0.0112</td>
</tr>
<tr>
<td>graph (Strategy 1)</td>
<td>0.0073</td>
<td>0.0088</td>
<td>0.0078</td>
</tr>
<tr>
<td>user-based</td>
<td>0.0061</td>
<td>0.0077</td>
<td>0.0068</td>
</tr>
<tr>
<td>item-based</td>
<td>0.0014</td>
<td>0.0023</td>
<td>0.0017</td>
</tr>
<tr>
<td>random sampling</td>
<td>0.0007</td>
<td>0.0007</td>
<td>0.0007</td>
</tr>
</tbody>
</table>

From the experimental evaluation of the proposed methods, we made some important observations. First, the proposed method provides a better quality of recommendation than the other methods do. The results show that the use of the relationships between items expressed by a graphical structure is beneficial. Second, in terms of the recommendations made, our proposed method provides better quality recommendations for items based on both the Top-5 and Top-10 recommendation results. In particular, the second strategy, which created the recommendation list by using the D'Hondt method, obtained the best quality. This outcome shows that the important thing is to reflect the diversity in the preferences of the target users. Purchased items within recommended items are different between Strategy 2 and the user-based CF. The former includes many types of artists or genres. For example, hip-hop musicians Usher and Jay-Z are included, as are famous Japanese singers such as “Naotaro Moriyam” and “Mr. Children”. Although the experimental results confirmed that these selected methods are suitable, the quality of recommendations they generated is still low to an extent. Therefore, some improvements are necessary to generate a continually high-quality Top-N list.

CONCLUSIONS AND REMARKS

In this paper, we proposed a recommender system using a graph-partitioning method for binary data without ratings. In the computational experiments, we illustrated that our proposed method outperforms other existing methods. In actual business applications, it is expected that our method will display a good performance.

Using the D'Hondt method to select a number of items from a subgraph has improved the performance of our method, because it can incorporate users’ preferences into the recommendation algorithm. Once it has done that, it is needed to balance the influence between other users’ actions and a particular user’s preference. To decide on and implement an optimal balance is difficult, because the optimal balance may vary depending on the item, the users, and so on.

One idea for improving the performance is learning how to select particular items from the subgraph. In the present version, we focus on strong relationships with existing purchased items independently. Incorporating a rule-based approach may create more steady recommendations. Incorporating these ideas, we would like to improve the performance of recommendations even further in subsequent works.

Notes

1) Data obtained from the 2006 Data Analysis Competition, which is sponsored by the Joint Association Study Group of Management Science (JASMAC).
2) http://glaros.dtc.umn.edu/gkhome/views/metis/

References