Re-Scaled Weight based Projected Graph with Rank Node

Haechan Jeong
Cho Chun Shik Graduate School of Mobility, KAIST
Daejeon, South Korea
haechan@kaist.ac.kr

Seungah Son
Cho Chun Shik Graduate School of Mobility, KAIST
Daejeon, South Korea
seungahson@kaist.ac.kr

ABSTRACT

This paper discusses algorithms for inferring missing information based on the given information between users, itemsets, and items. It consists of two tasks: task1, which utilizes user-itemset data, and task2, which utilizes itemset-item data. By using a bipartite graph, we obtain the projected graph and the adjacency matrix S. Then, we calculate re-scaled weights based on the probability transition matrix $A^{(i)}$ corresponding to rank i. With the obtained information, we perform task1 and task2. Through the analysis of the problem situations for task1 and task2, similar but appropriate algorithms are devised for each.

1 INTRODUCTION

The problem we want to solve in task1 is the following:

- GIVEN: Information about which itemset a user has chosen.
- FIND: Predict which itemset a specific user will choose.
- to MAXIMIZE: Accuracy, calculated as the number of correct predictions divided by the number of test cases.

The problem we want to solve in task2 is the following:

- GIVEN: Information about the items that fit a certain itemset.
- FIND: The top 100 list of items that fit this imperfect itemset.
- to MAXIMIZE: Average accuracy based on the presence of missing items and average rank with respect to the ground truth.

The contributions of this project are the following:

- Our proposed "Re-Scaled Weight based Projected Graph with Rank Node" has been developed innovatively.
- We provided variations in the methods according to the tasks to present efficient algorithms for each situation.
- Appropriate data structures were used to efficiently process large datasets.
- Detailed analysis of the problem situations for each task helped reduce computational costs.

2 PROPOSED METHOD

The overall process, as can be seen in Fig 1, involves constructing a bipartite graph and projecting it to obtain a projected graph and adjacency matrix S. Re-scaled weights are computed based on the probability transition matrix $A^{(}i)$ corresponding to the selected rank i.

Our proposed method is as follows :
A. Projection
Based on the given dataset, a bipartite graph can be created to represent sets on the left and right sides.

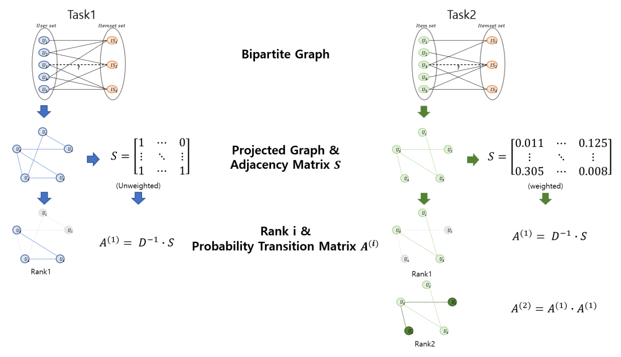

Task1
Task2
User set
Itemset set
Item set
Bipartite Graph
$S = \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}$
(Unweighted)
Projected Graph & Adjacency Matrix S
$S = \begin{bmatrix} 0.011 & \cdots & 0.125 \\ \vdots & \ddots & \vdots \\ 0.305 & \cdots & 0.008 \end{bmatrix}$
(weighted)
$A^{(1)} = D^{-1} \cdot S$
Rank i & Probability Transition Matrix $A^{(i)}$
$A^{(1)} = D^{-1} \cdot S$
Rank1
Rank1
$A^{(2)} = A^{(1)} \cdot A^{(1)}$
Rank2

Figure 1: The Overall Process

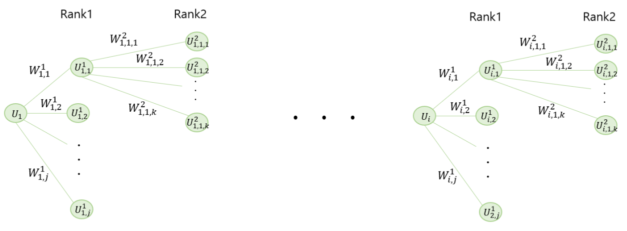
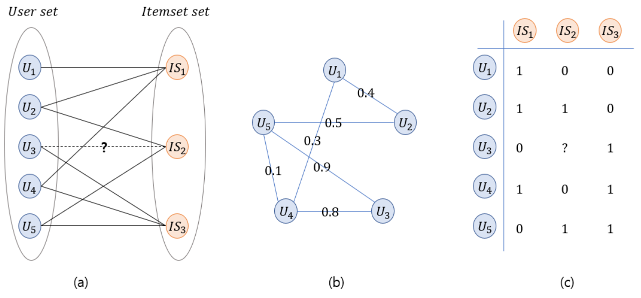


Figure 2: The Overall Process



	IS_1	IS_2	IS_3
U_1	1	0	0
U_2	1	1	0
U_3	0	?	1
U_4	1	0	1
U_5	0	1	1

Figure 3: The Overall Process

-Task 1
In order to predict whether a user will select or not select an itemset based on their similarity with other users and the selection information of those users, a bipartite graph is constructed as follows: the user node set is located on the left side of the bipartite graph, and the itemset node set is located on the right side. The edges representing the relationship between these two sets are constructed based on the 'user-itemset training' dataset.

-Task 2

Items with similar attributes are more likely to be included in similar itemsets. When given information about whether a certain item is included in a specific itemset, it is necessary to measure the mutual similarity between this item and other items. Therefore, a bipartite graph is constructed as follows: the item node set is located on the left side, and the itemset node set is located on the right side. The edges are constructed based on the 'itemset-item training' dataset.

However, this bipartite graph represents a large number of connections between nodes, making it difficult to comprehend at a glance. Therefore, in order to gain insights from this data and utilize it to solve the given problem, projection is used to compress it into a single population. For both Task 1 and Task 2, projecting the set located on the left side yields a projected graph.

B. Adjacency Matrix

The connectivity between a node i and another node j is represented by the element $s_{i,j}$ in the adjacency matrix. In the case of an unweighted adjacency matrix, $s_{i,j}$ is represented as either 0 or 1, i.e., binary. In the case of a weighted adjacency matrix, $s_{i,j}$ is represented by the weight value determined by a weight function.

-Task 1

Since weights are not considered, an unweighted adjacency matrix is obtained.

-Task 2

The projected graph contains much less information than the original bipartite graph, leading to a loss of some information from the original graph. To address this issue, an appropriate weight function is used to generate a weighted adjacency matrix. In Task 2, the weight function used is the Jaccard similarity. Jaccard similarity is a measure that reflects the similarity between two nodes, making it suitable for Task 2.

C. Rank

From the projection and weights, we can gather direct and indirect association information between nodes. The node we want to predict is called the "target node," and the other nodes are referred to as "surrounding nodes." For example, in Fig 3, if we consider U3 as the target node, U4 and U5 are directly connected to U3, so their selection information directly influences the prediction of U3's selection. On the other hand, U1 and U2 do not have a direct influence on U3 but have a relationship with U4 and U5, so they might indirectly influence U3.

For a target node, the set of surrounding nodes that have i links is defined as rank i. For example, U4-U3 and U5-U3 have one link, so U4 and U5 can be considered rank 1 nodes with respect to U3. U1-U4-U3, U1-U5-U3, and U2-U5-U3 have two links, so U1 and U2 are rank 2 nodes.

However, if the dataset being handled is large, the number of nodes in the projected graph can be significantly high. Additionally, if the nodes have closely related similar relationships, the interconnection between nodes can become very complex. Therefore, determining the number of ranks to consider, based on a thorough analysis of the problem situation, is crucial as considering too many ranks can lead to a significant increase in computational costs relative to accuracy. We have analyzed the problem situations for Task 1 and Task 2.

-Task 1

Only consider rank 1 nodes (directly connected nodes in the projected graph). When predicting the selection of the target node, it is sufficient to consider the selection information of directly influencing nodes as well as indirectly connected nodes with relatively lower relevance. Considering the selection information of indirectly connected nodes may negatively impact accuracy. In the progress report, a method considering up to rank 2 was suggested. However, later experiments revealed that considering only rank 1 resulted in similar accuracy with significant differences in computational costs.

-Task 2

Consider both rank 1 and rank 2 nodes. To find a list of 100 candidate items that can be missing from an itemset, a broad range of nodes need to be explored. Additionally, since nodes in the projected graph are closely connected based on similar characteristics, it is necessary to explore a sufficient number of ranks. Theoretically, it may require exploring more ranks beyond rank 2, but based on the "itemset item training dataset," it was observed that all nodes had more than 100 neighboring nodes when exploring up to rank 2. Therefore, considering computational costs, only rank 2 is explored.

D. Transition Probability

In the given fig 2, to calculate the re-scaled weight between the target node I_i and the rank 2 node $I^2_{1,1,1}$, we need to compute $W^1_{1,1} \times W^2_{1,1,1}$. ($W^1_{1,1}$ represents the weight between the target node and rank 1 nodes, and $W^2_{1,1,1}$ represents the weight between rank 1 nodes and rank 2 nodes.) Performing such computations for all nodes is computationally inefficient.

In 'GraRep: Learning Graph Representations with Global Structural Information', the probability transition matrix A is introduced. It is defined as $A = D^{-1} \cdot S$, where S is the adjacency matrix and D is the degree matrix. The element $a_{i,j}$ of A represents the probability of transitioning from node i to node j. Based on the definition, A can be seen as a row-normalized and re-scaled version of S. Additionally, $A^k = AA \ldots A$ can be calculated, where $a^k_{i,j}$ represents the probability of transitioning from node i to node j in k steps.

Inspired by 'GraRep: Learning Graph Representations with Global Structural Information', we can easily obtain the re-scaled weight between the target node and nodes in rank i by introducing the probability transition matrix A. By calculating A^2, we can obtain the values corresponding to the target node and rank 2 nodes, simplifying the process of obtaining the re-scaled weight. Furthermore, the calculation of A^2 can be performed as a matrix operation, making it efficient in terms of time cost.

-Task 1

Since only rank 1 nodes are considered, the re-scaled weight matrix is $A^{(1)}$.

-Task 2

Considering both rank 1 and rank 2 nodes, if we pre-calculate A and A^2, we can easily obtain the re-scaled weight between the target node and rank 1 or 2 nodes by searching for the corresponding elements. The actual implementation code is designed to minimize memory usage by leveraging the mechanism of matrix operations to reduce the computational cost.

Now, each task is performed with information obtained through the process A to D.

-Task 1

To predict which itemset the target node selected, we can calculate the total influence c_k^{IS} coming from the neighboring nodes of the target node k using the information of whether the rank 1 nodes selected that itemset and the re-scaled weights. It can be computed as follows:

$$c_k^{IS} = \sum c^{IS} ja(j,k) \tag{1}$$

Here, j represents one of the nodes included in rank 1, and $c^{IS} j$ represents the information of whether the neighboring node j selected the itemset, taking a value of 0 or 1. $a(j,k)$ denotes the re-scaled weight between the neighboring node j and the target node k. For example, in Fig 3, if U4 did not select IS2 with a weight of 0.8, and U5 selected IS2 with a weight of 0.9, the influence coming from rank 1 would be $0 \times 0.8 + 1 \times 0.9$. The prediction of whether the target node k will select itemset IS can be obtained using the following equation:

$$f(c_k^{IS}) = \begin{cases} 0, & \text{if } c_k^{IS} < \epsilon \\ 1, & \text{if } c_k^{IS} \geq \epsilon \end{cases} \tag{2}$$

Comparing c_k^{IS} with a pre-selected threshold, if the value is large, we can predict that the target node k will select a specific itemset. Conversely, if the value does not reach the threshold, we can predict that the target node will not select a specific itemset. The optimal threshold value can be obtained by comparing the prediction results with the validation dataset.

-Task 2

Assuming there are nodes $I_1, I_2, ..., I_i$ that have information about being included in a specific itemset, we can find the nodes included in rank 1 and 2 using these nodes as target nodes. Then, we obtain the re-scaled weights between the target nodes and their neighboring nodes using A and A^2. We repeat this process until I_i, and then we sort the nodes in descending order of weights and obtain information about the top 100 nodes. Since nodes with higher similarity to the target nodes have larger re-scaled weights, the final selected 100 items are likely to be suitable for the mentioned itemset, as initially stated.

3 EXPERIMENTS

A. Finding Optimal Threshold A. Finding Optimal Threshold

Figure 2 (a) shows the graph of accuracy with respect to the proposed method using the validation dataset for different thresholds. As the threshold increases, there is a tendency for the ratio of Negative Positive to increase logarithmically. As the threshold increases, the False Negative becomes significantly larger than False Positive, resulting in a rapid decrease in accuracy. Figure 2 (b) presents the accuracy analysis for the threshold range [0.0001, 0.002]. From this, the highest accuracy of 0.788 was achieved at a threshold of 0.00107.

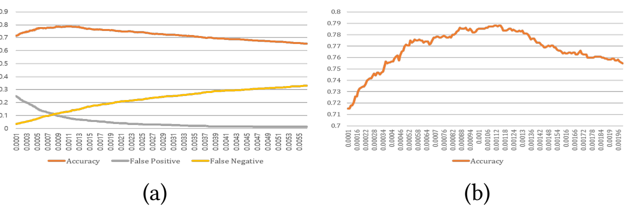


Figure 4: (a) The result of the proposed method (b) Accuracy to find Optimal threshold

B. Analysis of the confusion matrix

Table 1 represents the results obtained by sampling 1140 data points from the validation dataset and classifying them into a confusion matrix. As shown in Table 1, false positives are relatively higher than false negatives. The sensitivity calculated from this confusion matrix is 0.9403, indicating that the model has a high recall and effectively identifies positive samples.

		True Label		
		Positive	Negative	Total
Predicted Label	Positive	630	286	916
	Negative	40	184	224
	Total	670	470	1140

Table 1: Confusion Matrix

4 CONCLUSIONS

The proposed method *Re-Scaled Weight based Projected Graph with Rank Node* has the following advantages:

The neighbor-based method may not provide accurate predictions for cases where neighbors are scarce, such as new users with limited history. On the other hand, the method proposed in this project considers not only rank1 but also rank2, thus achieving high prediction accuracy even in this case.

Secondly, instead of relying on traditional data structures like dataframes, we explored the use of sorted arrays and binary search to process large datasets more efficiently. This approach proved to be highly effective and resulted in improved computational speed and resource utilization.

Furthermore, we conducted a detailed analysis of the problem situations for each task, which helped us reduce computational costs. By gaining a deeper understanding of the underlying challenges and making informed decisions, we were able to streamline the processes and enhance overall efficiency.

The following sections describe the limitations and potential improvements of the proposed method based on experimental results. These suggestions will be incorporated to improve the accuracy of the proposed method.

- *Accuracy is threshold-sensitive*: The value of c_k^{IS} is too small because the weight obtained from the Jaccard similarity is small. The average of the c_k^{IS} values obtained with $f(c_k^{IS}) = 1$ is 0.00184, while the average value of $f(c_k^{IS}) = 0$ is 0.00049, confirming that there was no significant difference. To solve this problem, we intend to modify the weight function or introduce a new weight function, which guarantees that c_k^{IS} derived from $f(c_k^{IS}) = 0$ and $f(c_k^{IS}) = 1$ gives a significant difference in the value.

- *There are far more false positives than false negatives*: This seems to be because the denominator is too small compared to the denominator in the normalization process. Therefore, we aim to significantly improve accuracy by modifying the normalization method to reduce the number of false positives.

REFERENCES

A APPENDIX

A.1 Labor Division

The team performed the following tasks

- Implementation of weight propagation with rank node [all]
- Experiments on the real data [Haechan]
- Comparison on weight propagation with rank node against Cosine Similarity method [Seungah]

A.2 Full disclosure wrt dissertations/projects

Haechan: He is not doing any project or dissertation related to this project: his thesis is on using Graph Neural Network into the telecommunications field.

Seungah: She is not doing any project or dissertation related to this project: her thesis is on applying reinforcement learning to robot soccer game.