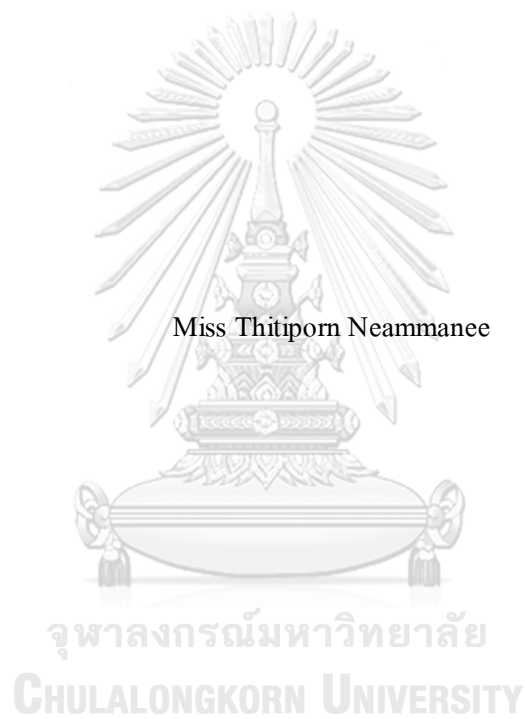


Considering Neighbor Projection on Neural based Recommender Systems



Miss Thitiporn Neammanee

A Thesis Submitted in Partial Fulfillment of the Requirements  
for the Degree of Master of Science in Computer Science and Information Technology

Department of Mathematics and Computer Science

FACULTY OF SCIENCE

Chulalongkorn University

Academic Year 2021

Copyright of Chulalongkorn University

ระบบแนะนำที่คำนึงถึงเพื่อน โดยใช้โครงข่ายประสาทเทียม



วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต  
สาขาวิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ ภาควิชาคณิตศาสตร์และวิทยาการ

คอมพิวเตอร์

คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2564

ลิขสิทธิ์ของจุฬาลงกรณ์มหาวิทยาลัย





# # 6172608223 : MAJOR COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

KEYWORD: Neural Collaborative Filtering, Rating conversion, Collaborative Filtering,  
Recommender System, neighbor's concept

Thitipom Neammanee : Considering Neighbor Projection on Neural based  
Recommender Systems. Advisor: Assoc. Prof. SARANYA MANEEROJ, Ph.D.

Now, CF is applied with a neural network to make the model more flexible and more accurate. Different neighbors have a different influence on the target user, and different users usually have different rating patterns. Therefore, the proposed method needs to consider two major issues when applying CF with a neural network: the similarity levels between the neighbors and the target user and the user's rating pattern conversion. Thus, the proposed method consists of three main modules to solve the issues mentioned above: rating conversion, similarity module uses, and prediction module. In the experiment, the proposed method is evaluated and compared with the current neural CF with friends and latent factor model on two types of datasets: real-world and synthetic datasets. In real-world datasets,  $N$  neighbors and all neighbors are evaluated to demonstrate the significance of the number of neighbors. Furthermore, the rating conversion module's performance is assessed by comparing the results of the proposed method with and without the rating conversion module. For the synthetic datasets, this work simulates the full rating matrix datasets and the partial rating matrix dataset to compare the effectiveness of using different types of distribution and dataset size. The experimental results demonstrate that the proposed method effectively outperformed the baselines utilizing ranking evaluation and prediction accuracy on real-world and synthetic datasets.

Field of Study:	Computer Science and Information Technology	Student's Signature .....
Academic Year:	2021	Advisor's Signature .....

## ACKNOWLEDGEMENTS

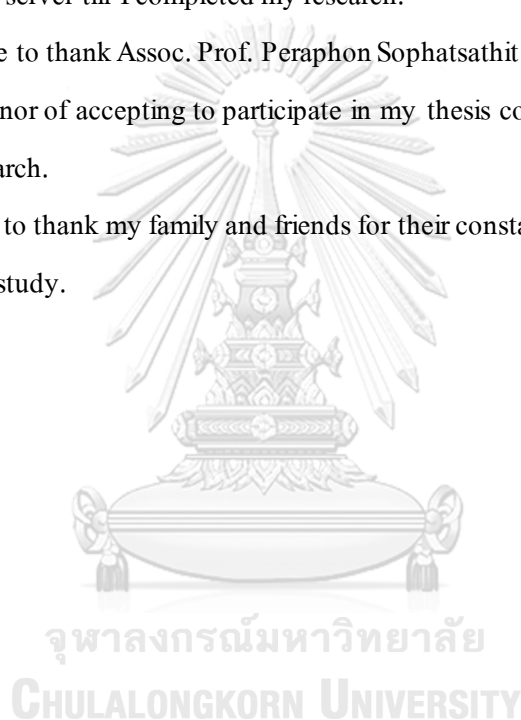
First of all, I would like to express my gratitude to my thesis advisor, Assoc. Prof. Dr. Saranya Maneeroj. Without her assistance, valuable guidance, and continuous encouragement in every step throughout my thesis, I would have never accomplished this work.

In addition, I would like to thank Prof. Dr. Atsuhiko Takasu at the National Institute of Informatics for being the advisor during the internship, giving continuous support, and providing constant help and the server till I completed my research.

I would like to thank Assoc. Prof. Peraphon Sophatsathit and Asst. Prof. Annupan Rodtook for giving me the honor of accepting to participate in my thesis committee and giving me valuable feedback on my research.

I also want to thank my family and friends for their constant support all over the time since the beginning of my study.

Thitiporn Neammanee



## TABLE OF CONTENTS

	<b>Page</b>
ABSTRACT (THAI) .....	iii
ABSTRACT (ENGLISH).....	iv
ACKNOWLEDGEMENTS.....	v
TABLE OF CONTENTS.....	vi
LIST OF TABLES .....	1
LIST OF FIGURES.....	1
CHAPTER 1 INTRODUCTION.....	2
CHAPTER 2 RELATED WORKS .....	6
2.1 COLLABORATIVE FILTERING.....	6
2.2 NEURAL CF WITHOUT NEIGHBORS' CONCEPT .....	7
2.3 NEURAL CF WITH NEIGHBORS' CONCEPT.....	11
2.4 RATING CONVERSION.....	16
2.5 REGION EMBEDDING.....	17
CHAPTER 3 PROPOSED METHOD.....	19
3.1 OVERVIEW OF PROPOSED METHOD.....	19
3.2 PROCESS OF PROPOSED METHOD .....	22
3.2.1 SELECTING NEIGHBORS.....	23
3.2.2 USER-ITEM EMBEDDING.....	23
3.2.3 CONVERTING NEIGHBOR INTO THE TARGET USER ASPECT .....	24
3.2.4 CREATING SIMILARITY .....	25
3.2.5 RATING PREDICTION.....	25

3.3 TIME COMPLEXITY OF PROPOSED METHOD .....	26
CHAPTER 4 EVALUATION.....	30
4.1 REAL-WORLD DATASETS .....	30
4.1.1 DATA PREPARATION.....	31
4.1.2 EXPERIMENTAL SETTINGS.....	32
4.1.3 EVALUATION METRICS.....	33
4.1.3.1 RANKING-BASED EVALUATION .....	33
4.1.3.2 PREDICTION ACCURACY .....	34
4.1.4 NEIGHBOR SELECTION.....	34
4.1.4.1 $N$ NEIGHBORS EXPERIMENTAL RESULTS .....	35
4.1.4.2 ALL NEIGHBORS EXPERIMENTAL RESULTS .....	45
4.2 SYNTATIC DATASET .....	46
4.2.1 DATA GENERATION SETTING.....	46
4.2.2 EXPERIMENTAL RESULTS .....	49
CHAPTER 5 DISCUSSIONS.....	54
5.1 PERFORMANCE OF SIMILARITY MODULE.....	54
5.2 PERFORMANCE OF RATING CONVERSION MODULE .....	55
5.3 NEIGHBOR SELECTION.....	56
5.4 DATA SIMULATION.....	57
5.5 COMPARING METHOD WITH LATENT FACTOR MODEL .....	58
CHAPTER 6 CONCLUSIONS .....	59
REFERENCES.....	62
VITA.....	66



## LIST OF TABLES

Table 1 Comparison of the related works with respect to the neighbor concept.....	16
Table 2 The number of records, users, and item in each dataset .....	31
Table 3 The number of records, users, and item in each dataset after preprocessing.....	32
Table 4 Comparison of the experimental results on all datasets using nDCG.....	37
Table 5 Comparison of the experimental results on all datasets using the HR .....	38
Table 6 Predicted accuracy experimental results .....	40
Table 7 Comparison of the experimental results in terms of percentage between the proposed method without rating conversion and the other methods using nDCG .....	42
Table 8 Comparison of the experimental results in terms of percentage between the proposed method without rating conversion and the other methods using HR .....	42
Table 9 The experimental results comparison between the other methods and the proposed method without rating conversion in terms of percentage using prediction accuracy metrics .....	43
Table 10 Comparison of the experimental results in terms of percentage between the proposed method with and without rating conversion using nDCG.....	44
Table 11 Comparison of the experimental results in terms of percentage between the proposed method with and without rating conversion using HR.....	44
Table 12 The experimental results comparison between the proposed method with and without rating conversion in terms of percentage using prediction accuracy metrics .....	44
Table 13 The experimental results of using N neighbor and all neighbor .....	45
Table 14 The parameters of synthetic data.....	48
Table 15 The nDCG experimental results of synthetic data.....	49
Table 16 The accuracy experimental results of synthetic data.....	51

## LIST OF FIGURES

Figure 1 an example of rating matrix .....	6
Figure 2 The example of user-item rating matrix and K-Nearest Neighbors rating matrix .....	12
Figure 3 Context-Word Region embedding Method.....	18
Figure 4 Context unit example.....	19
Figure 5 Similarity among users example .....	20
Figure 6 Rating conversion issue example.....	21
Figure 7 The proposed model architecture.....	22
Figure 8 The example of neighbors selection .....	23
Figure 9 The proposed method neural network architecture.....	26
Figure 10 Experimental results of Movielens 1M dataset for the proposed method varying the embedding size and neighbors. ....	36
Figure 11 Experimental results graph comparison on all datasets using the nDCG and HR.....	39
Figure 12 Similarity values between the neighbors and the target user of three datasets.....	46
Figure 13 The example of synthetic datasets.....	47
Figure 14 The example of Rating range issue .....	55
Figure 15 The example of rating normalization problem .....	56
Figure 16 The example of comparing k ranking .....	58

## CHAPTER 1 INTRODUCTION

A Matrix Factorization (MF) [5-7] is the most popular model for RS and applied with the Collaborative Filtering technique. A Matrix Factorization approach was proposed by Koren et al [6], which decomposes a matrix into a lower dimension. It means that a Matrix Factorization is used to predict the target item predicted rating of the target user. Usually, the rating of the target user on the items can be viewed in the form of a rating matrix. This rating matrix can be decomposed into user and item latent feature vectors, representing user profiles and item characteristics.

The Matrix Factorization technique can be utilized to latent feature extraction methods such as the Singular Value Decomposition (SVD) [8], Principal Component Analysis (PCA) [9], and Latent Dirichlet Allocation (LDA) [10]. In order to perform the rating prediction using the MF technique, a user representation and an item representation are combined using an inner product in latent factor space. The prediction equation of MF concept has shown in Equation 1.

$$r_{ui} = p_u \cdot q_i \quad (1)$$

where  $r_{ui}$  denotes the predicted rating of target user  $u$ 's on item  $i$ ,  $p_u$  is latent factor of user  $u$ , and  $q_i$  is latent factor of item  $i$ . Now, Neural Network (NN) is publicly utilized in word embedding and Recommender Systems, such as Convolutional Neural Network (CNN) [11], Recurrence Neural Network (RNN) [12], Long-Short Term Memory (LSTM) [13-15]. Using a neural network makes the models more accurate and flexible. There are three layers of a basic neural network: the input layer, hidden layer, and output layer, which contain many nodes or neurons in each layer. All of the input features are represented as nodes in the input layer. In order to compute the hidden layer, all input nodes are computed with weights and bias. Lastly, the outputs from the hidden layer nodes are adapted to predict or classify results.

The Neural Collaborative Filtering (NCF) [16], the Outer Product-based Neural Collaborative Filtering (ONCF) [17], and temporal CNN for reviews based on recommender system (TCR) [18] are the examples of research that utilizing the Collaborative Filtering with a neural network. NCF applied the multilayer perceptron (MLP) neural network with the generalized MF

(GMF) models to combine the users' information into the users' ratings. ONCF aims to enhance the NCF using CNN instead of MLP. In contrast, TCR added a time component into the CNN model, which helps to modify the importance of users in chronological order. However, all NCF, ONCF, and TCR learn user-item interactions employing users' ratings. These models do not consider how much each user influences the target user as the concept of neighbors in the CF technique.

There are four examples of applying the neighbors' concept of the CF technique into neural models, A Neural Collaborative Filtering Model with Interaction-based Neighborhood (NNCF) [19], Recommendation Based on BP Neural Network with Attention Mechanism (BPAM) [20], the collaborative memory network for recommendation systems (CMN) [21] and the social attentional memory network: modeling aspect and friend-level differences in recommendation (SAMN) [22]. The neighbors' ratings and the similarity between the neighbor and the target user are the keys to leveraging the neighbor's concept in a Collaborative Filtering model. The NNCF explicitly used neighbors in their model, but the importance levels of neighbors are not concerned as in the CF concept. The BPAM proposes utilizing the influence of the target user on the neighbors by performing a similarity between the neighbor and the target user. However, it did not compute or utilize the neighbors' ratings, which are important in CF's technique.

Both CMN and SAMN generate a target user's profile, which combines all neighbor's influences. Afterward, the target user's profiles and the target item are used to perform a prediction. These neighbors' influences, which are the similarity between the neighbor and the target user, are a combination of the neighbor's embedding and the similarity between the neighbors and the target user. The prediction process is the significant difference between CMN and SAMN. After obtaining the neighbor's influences, these influences are learned with the target user embedding and the target item embedding through a neural network of the CMN model. In comparison, SAMN intergrades the neighbor's influence with the target user embedding to perform the target user's profile. After that, this target user's profile is multiplied with the item embedding using dot product to create the predicted rating as in the matrix factorization concept. The CF's prediction equation integrates the neighbor's ratings and the similarity between the neighbor and the target user using a weighted average. Due to the neighbor's ratings are not applied or generated in both current NCFs with friends prediction process. Therefore, CMN and SAMN still cannot simulate the CF prediction process using a neural network.

In addition to the two keys, another issue in the collaborative filtering technique is that different users have different rating ranges. Many current works use neighbor's ratings to predict the target user rating directly without converting the rating range into the target user aspect. These current works used different rating ranges to predict the items' ratings for the target user. Thus, using the neighbor's ratings directly can lead to the rating conversion issue and inaccurate prediction.

In this work, two main issues need to be considered: the rating conversion and the similarity among users. Under these two main issues, three main components of CF's prediction are achieved: the similarity level between users, the neighbor's rating, and the rating conversion. Thus, the proposed method consists of three modules: the rating conversion module, the similarity module, and the prediction module. The rating conversion module aims to achieve the neighbors' ratings in the target user's aspect, which is one issue that needs to be concerned. In order to perform the neighbor's ratings in the target user aspect, neighbors' vectors are projected with the target user aspect. Afterward, these neighbors' vectors and the item's vector are integrated utilizing the MF technique to receive the neighbors' ratings in the target user aspect. The similarity module performs the neighbors' attention to capture the similarity levels between the neighbors and the target user. These attentions can be obtained by applying a dot product between a vector of the neighbor and the target user. Both the rating conversion and similarity module results are combined in the prediction module. The prediction module uses the neighbor's rating in the target user aspect and the similarity levels to compute the target user's predicted rating using a weighted average, imitating the CF's prediction equation, where the similarities are used as weights. In the experiment, there are four evaluation objectives:

- 1) To examine whether the proposed method outperforms the current NCFs with friends and the latent factor models.
- 2) To evaluate the effectiveness of the rating conversion module in the proposed method
- 3) To evaluate the number of neighbors performance on real-world datasets
- 4) To compare the efficiency of the different rating distributions and dataset size in the synthetic dataset.

There are two types of evaluation metrics, which used to evaluate the proposed method: ranking-based evaluation, including normalized discounted cumulative gain (nDCG) and hit ratio

(HR); and prediction accuracy metrics, including precision, recall, and root mean square error (RMSE). The proposed method are evaluated on two types of dataset, which are real-world and synthetic datasets. There are two types of neighbors selection to evaluate the effect of the number of neighbors in the real-world dataset:  $N$  neighbors and all neighbors. Moreover, the efficiency of the rating conversion module is evaluated in this work. Therefore, the proposed method is compared with the proposed method without rating conversion. The synthetic datasets are generated and used to evaluate the impact of data distribution and the dataset size on an ideal rating matrix. Besides, the partial rating matrix is generated to simulate the real-world dataset with the normal rating distribution and compare it with the ideal rating matrix. The experimental results of the proposed method are compared with the results of current NCFs with friends methods and the two latent factor models on both types of datasets. The outcomes show that the proposed method significantly outperforms when compared with all baselines. Thus, the considering similarity and the rating conversion of neighbors on neural collaborative filtering has three contributions as follows.

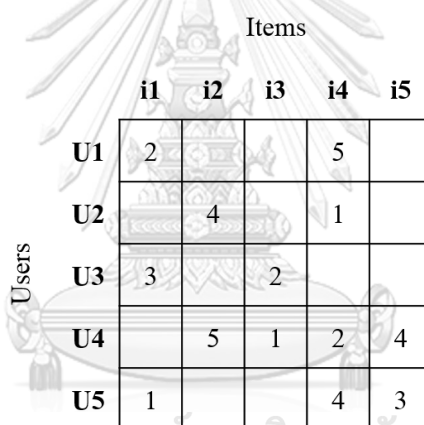
- The rating conversion module transforms the ratings from the neighbor's perspective to the target user's perspective, which has not been considered. It can solve the rating range issue by projecting the neighbor's characteristics into the target user's perspective, which can transform the neighbor's rating range to the target user rating range.
- The similarity module can directly capture the similarity levels between the neighbor and the target user by utilizing the neighbors' attention. The neighbors' attentions are performed by applying a dot product between the neighbor's representation vector and the target user profile. Afterward, the neighbors' attentions are normalized by using a softmax function. The users who have a similarity value more than zero as a neighbor. Therefore, the similarity levels range is in  $[0,1]$ , which is in the same range as the CF's similarity.
- There are three main components to achieve applying a neural network with the collaborative filtering technique in this work: the similarity between the neighbor and the target user, the neighbor's rating, and the rating conversion. However, the existing models concern only the similarity values or only neighbor's ratings. In comparison, the proposed method considers all three main components.

## CHAPTER 2 RELATED WORKS

In this section, the researches which relate to the proposed method are described and discussed. Begin with the principle of collaborative filtering, neural CF without neighbor's concept, neural CF with neighbor's concept, rating conversion, and region embedding.

### 2.1 COLLABORATIVE FILTERING

This work focuses on the collaborative filtering technique, one of the famous techniques in the recommender system. The CF technique captures the user-item interaction in the past in a user-item matrix form called a rating matrix. Figure 1 is an example of a rating matrix. The number in matrix elements is the user's rating on an item. The blank elements refer to the item that the user has never rated before.



		Items				
		i1	i2	i3	i4	i5
Users	U1	2			5	
	U2		4		1	
	U3	3		2		
	U4		5	1	2	4
	U5	1			4	3

Figure 1 an example of rating matrix

The users who have rated the same item in the past are called co-rate users. For example, users U1, U3, and U5, in Figure 1, are co-rate users because all of them have rated the same item, which is item i1. The users who have rated the target item are called raters. Suppose user U2 is the target user and item i4 is the target item. User U1, U4, and U5 have rate item i4. Therefore, users U1, U4, and U5 are called raters. The CF technique is based on the assumption that similar users will have similar preferences. The users who have a similar preference with the target user are called neighbors. This technique use neighbors' ratings ( $r_{n,j}$ ) to predict the rating for the target user as the CF's prediction equation:

$$r(i, j) = \frac{\sum_{n \in N} \text{sim}(i, n) \cdot r_{n, j}}{\sum_{n \in N} |\text{sim}(i, n)|} \quad (2)$$

where  $r(i, j)$ ,  $\text{sim}(i, n)$  denote the predicted rating of item  $j$  for the target user  $i$  and the similarity between the target user  $i$  and user  $n$ , respectively.  $N$  is the set of neighbors,  $r_{n, j}$  is the rating of neighbor  $n$  given to item  $j$ . From Equation 2, another factor employed to perform the target user rating is the similarity between the neighbor and the target user. It is used to indicate how similar the user is to the target user. This similarity value can be calculated by applying many techniques such as cosine similarity, Pearson's correlation, et cetera. Furthermore, the target user's neighbor's similarity to the target user is often used to determine the neighbors. The higher the similarity value, the user is more similar to the target user.

## 2.2 NEURAL CF WITHOUT NEIGHBORS' CONCEPT

A few years ago, the neural network was utilized with the collaborative filtering technique. The users and items embedding vectors are used as inputs. These embedding vectors are users and items representation vectors that learn by employing a neural network. To perform the users and items embedding, the user and item vectors are initialized. Then, the matrix factorization concept is used to create a predicted rating ( $\hat{r}_{p_u, q_i}$ ):

$$\hat{r}_{p_u, q_i} = p_u \odot q_i \quad (3)$$

where  $\odot$  denotes the element-wise product between two vectors.  $p_u$  and  $q_i$  are the user and the item embedding, respectively. The target user's actual rating values on the target item are labeled for comparing error and applied the backpropagation method. The number of layers and the hidden layer dimensions in this neural network must be adapted suitably with the data and input dimensions. Each initialized user embedding and item embedding are improved according to each user profile and item characteristics in the backpropagation process. After performing users and items embedding vectors, these vectors are used to learn the users' interaction using different types of neural network. NCF [16], ONCF [17], and temporal CNN for reviews based on the RS (TCR) [18] are examples of the NCF without neighbors concept.



NCF [16] proposed the GMF model, the MLP model, and the integration of the Generalized Matrix Factorization and the Multi-Layer Perceptron called the neural MF (NeuMF) model. The input of NCF model is the one-hot vector of the users, which is user's interaction on the item. The GMF model is the matrix factorization technique, which employing an element-wise product between the user and item vector to perform the predicted rating ( $\varphi^{GMF}$ ):

$$\varphi^{GMF} = a_{out}(w_{out}^T(p_u \odot q_i) + b_{out}) \quad (4)$$

where  $a_{out}$ ,  $w_{out}$  and  $b_{out}$  are the activation function, a matrix of edge weights, and bias of the output layer, respectively. The outcome of the GMF model is the predicted rating of the target user on the target item. The MLP model also uses user and item vectors. The user and item vectors are combined using concatenation to learn the interaction between users and items. The concatenation result is called concatenated vectors. After that, these concatenated vectors ( $z$ ) are used as input of NN and to learn the interaction between users and items through the network:

$$z = \begin{bmatrix} p_u \\ q_i \end{bmatrix}, \quad (5)$$

$$\varphi^{MLP} = a_{out}(w_{out}^T(a_L(w_L^T(a_{L-1}(w_{L-1}^T(\dots a_2(w_2^T(a_1(w_1^T z + b_1) + b_2) \dots)))) + b_{L-1}) + b_L) + b_{out})$$

where  $a_L$  denotes the activation function of the  $L$ -th hidden layer,  $w_L$  is the  $L$ -th weight vector of each hidden layer, and  $\varphi^{MLP}$  is the interaction vector which is the result from MLP network. The NeuMF model combines the GMF model and the MLP model together to integrate the user's interactions with the user's ratings. The result of GMF and MLP are the predicted rating value and the interaction vectors, respectively. Both results are merged into one vector using a concatenation operation in each user and item pair. Then, these concatenated vectors are learned through the output layer to perform the predicted rating value. In order to compare the prediction errors, the target user's actual ratings on the target item are labeled:

$$\hat{r}_{ui} = \sigma(h^T \begin{bmatrix} \varphi^{GMF} \\ \varphi^{MLP} \end{bmatrix}) \quad (6)$$

where  $h$  and  $\sigma$  are weight and activation function of the NeuMF network in output layer.  $\hat{r}_{ui}$  denotes the predicted rating of the target user  $u$  on the target item  $i$ . Therefore, this work propose a framework that learns the user's ratings and user's interactions, which applying only a matrix factorization. However, there is no neighbor's concept of CF technique, which uses the neighbor's preference to perform item recommendations.

ONCF [17] proposed the outer product based NCF framework, which adapts the NCF [16] model by applying CNN instead of MLP to learn the users' interactions. ONCF consist of three main steps: user-item embedding, Interaction map, and rating prediction. The input of the user-item embedding is one-hot encoding, which represents user-item features such as ID, user gender, item category. In order to perform the user-item embedding, the one-hot encoding of user and item is multiplied with the feature embedding matrix. After that user embedding ( $p_u$ ) is combine with the item embedding ( $q_i$ ) using an outer product operation to obtain the interaction map ( $E$ ).

$$E = p_u \otimes q_i \quad (7)$$

where  $E \in \mathbb{R}^{K \times K}$ ,  $K$  denotes the embedding size. In order to perform the predicted rating, the interaction map is extracted proper signals by employing a CNN to decrease a dimension until  $1 \times 1$ . It means the CNN layer is used to learn and predict the target user rating. However, ONCF employs only the user's interaction without using neighbors and the similarity levels between users, which is the CF's concept's main idea.

Due to user preference changes over time, the TCR model utilizes a CNN model with a time model. The TCR model consists of the user network and the item network, which execute the same process independently. This work utilizes reviews from the users to extract the user profiles and item characteristics using the embedding method [23, 24]. Afterward, Both user profiles and item characteristics are used as the input of a convolutional layer and max-pooling layer:

$$c_j^i = a(V * K_i + b_i) \quad (8)$$

where  $c_j^i$  denotes the outcome of the  $j^{th}$  convolution of the  $i^{th}$  layer,  $a$  is a activation function,  $*$  is denotes a convolution operation,  $V$  and  $K_i$  are the input and the convolution kernel size of the  $i^{th}$  convolutional layer. Then, the max-pooling operation is applied to the result obtained from each convolutional layer:

$$z_i = \max\{c_1^i, c_2^i, \dots, c_k^i\}, \quad (9)$$

$$z_{out} = \{z_1^i, z_2^i, \dots, z_n^i\} \quad (10)$$

where  $k$  denotes the convolutional kernel size,  $n$  is the number of pooling in the  $i^{th}$  layer.  $z_{out}$  is the output of the pooling layer. Because the user preference shifts over the time, the time model ( $w_k$ ) is proposed to reduce the role of the review  $k$ .

$$W_k = \frac{1}{T_c - T_k + b} \quad (11)$$

where  $T_c$  and  $T_k$  are the current time and the comment time of the review  $k$ .  $b$  is a bias which can prevent the error when  $T_c$  equal to  $T_k$ . Next, the results from pooling layer and the time model are merged using the concatenation operation. Then, the concatenated vectors are fed into the fully connected layer:

$$F = a((z_{out} \odot W_{time})w_F + b) \quad (12)$$

where  $W_{time}$  denotes a time weight matrix for the time model,  $w_F$  is the weight vector of the fully connected layer, and  $b$  is a bias.  $F$  is the output from the fully connected layer, which is the user profile vector ( $F_u$ ) in the user network and item characteristic ( $F_i$ ) vector in item network. The user profiles and item characteristics are used in the prediction step to perform the predicted rating vector using the matrix factorization technique (Equation 1).

However, the reviews can represent a user preference but cannot directly describe a user characteristic. The comments that cannot represent a user characteristic such as “fast delivery,” “Good quality bag,” and “Shoes are normally small on me so I went a size up and they fit great.” Moreover, the neighbors and the similarity between users, a key CF technique, are not considered in this framework.

### 2.3 NEURAL CF WITH NEIGHBORS' CONCEPT

Recently, there are some researchers have utilized the neighbor's concept with a neural network such as NNCF [19], BPAM [20], the CMN [21], and SAMN [22]. NNCF aims to combine the neighbors' information into the NCF model using concatenating operation. There are three steps to the NNCF method: user and item embedding, component integration, and rating prediction. In order to create a user and item embedding, the one-hot vectors of the target user, target item, the neighbors of neighbor (user neighbor), and the item that is similar to the target item (item neighbor) are used as input of the model. These one-hot vectors are applied with a concatenation-based look up layer to transform one-hot vectors into latent vectors. Therefore, the results of this step are the target user embedding vector, target item embedding vector, user neighbors' embedding matrix, and item neighbors' embedding matrix. In order to integrate all components, the target user and the target item are combined using a matrix factorization concept to create the predicted rating. Both user neighbors' embedding matrix and item neighbors' embedding matrix are transformed using convolution and a max-pooling operation to produce a new feature and capture the most important feature. The max-pooling operation results are the user neighbors' embedding vector and item neighbors' embedding vector. The predicted rating, the user neighbors' embedding, and item neighbors' embedding are integrated using concatenation operation. In order to learn the rating and neighbors' information, the concatenated vector is learned by using MLP to perform the predicted rating with the neighbors' information. In the component integration step, there is no computing and combining the similarity between the neighbor and the target user, which is the key of the CF technique. Therefore the neighbors of the target user are equally important, which can make the incorrect prediction.

BPAM proposed capturing neighbors' impact to predict the missing rating in the user-item rating matrix. BPAM consists of three main steps: K-Nearest Neighbors (KNN) rating matrix

construction, Data preprocessing, Attention mechanism. Firstly, the user's interactions on items are converted into the user-item rating matrix. To select the KNN rating matrix, K neighbors are selected by applying the cosine similarity between users. Suppose there are ten users and ten items in the dataset, and K is set to four. The top four users who are similar to the target user are selected as neighbors. Therefore, the KNN rating matrix is a submatrix with ten-by-five dimensions. Each element of the matrix is a rating of a user on an item. The last column is the rating of the target user on an item. Figure 2 shows an example of the user-item rating matrix and KNN rating matrix.

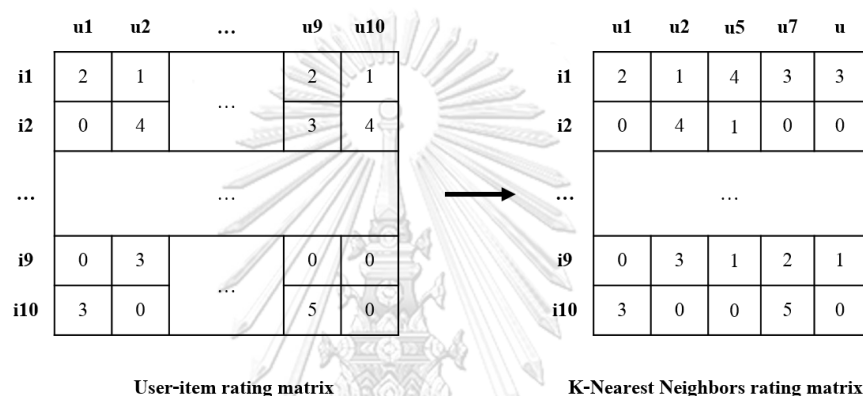


Figure 2 The example of user-item rating matrix and K-Nearest Neighbors rating matrix

After the KNN rating matrix of the target user is obtained, this KNN rating matrix is adjusted in the data preprocessing step. The zero elements of each neighbor are substituted by the mean rating of that neighbor. Finally, the local and global attention weights are created and used to perform the predicted rating. The local weight is the impact of the target user on the neighbor, which is a similarity between the neighbor and the target user. The global weight is the impact of the target user on all neighbors of the target user, which is performed by using two layers neural network. Local and global attention weights are combined and trained to perform the predicted rating. In order to apply a neural network with the CF neighbor's concept, there are two components to utilize in the model: the neighbor's rating and the similarity between the neighbor and the target user. Although this work creates and applies the similarity between the neighbor and the target user, the neighbor's rating is not performed or used in the prediction process as the CF prediction equation.

After the neural network is applied in the CF technique. Recently, some researchers applied the concept of neighbors into a neural network such as the Collaborative Memory Network for Recommendation Systems [21], and Social Attentional Memory Network: Modeling Aspect- and Friend-level Differences in Recommendation [22]. CMN [21] proposes capturing the user's similarity and enhancing the neighbor component into a neural network attention mechanism. This work consists of three steps, which are embedding, neighborhood attention, and rating prediction. First, the user interactions are represented as a vector and perform a user preference ( $q_{uiv}$ ).

$$q_{uiv} = m_u \cdot m_v + e_i \cdot m_v; \forall v \in N(i) \quad (13)$$

where  $m_u$ ,  $m_v$ , and  $e_i$  are the embedding vectors of target  $u$  user, neighbor  $v$ , and item  $i$ . In order to create the new target user representation, the neighbor component and the user embedding are combined. The neighbor component is the final neighbor's representation, which applies the attention weights. These attention weights infer the neighbor's importance using Equation 14 and fused with the neighbor embedding in a neural network layer to receive the final neighbor's representation:

$$o_{ui} = \left( \sum_{v \in N(i)} \frac{\exp(q_{uiv})}{\sum_{k \in N(i)} \exp(q_{uik})} \right) \cdot c_v \quad (14)$$

where  $c_v$  is another neighbor embedding vector called external memory. Finally, the target user's rating score is performed via the matrix factorization concept by employing the target user's and the target item's representation. Then, the final neighbor's representation and the target user's rating scores are learned utilizing a skip-connections neural network:

$$r_{ui} = v\phi(U(m_u \odot e_i) + W o_{ui} + b) \quad (15)$$

where  $r_{ui}$  denotes the predicted rating score of the target user  $u$  on the target item  $i$ .  $U$ ,  $W$ ,  $V$ , and  $b$  are learned parameters. The CF's prediction equation applies a weighted average using the

similarity levels between users and the neighbor's rating scores. However, CMN utilizes the neighbors' representations to combine with the target user rating scores instead of the similarity level as in the CF concept.

Social Attentional Memory Network: Modeling Aspect and Friend-level Differences in Recommendation proposed (SAMN) [20] proposes neighbor prioritizing, which considers the level of user preference. Because each neighbor of the target user has a different preference on the different target item. Thus, Chen et al. [20] decided to perform a user profile more accurately by combining a neighbors' influence into the target user embedding. First, a user and item are represented as user-item embeddings. The target user embedding ( $u_i$ ) and their neighbors embedding are fused through Equation 16 to perform an attention vector between the neighbor and the target user.

$$s = \frac{u_i \odot u_{(i,l)}}{\|u_i\| \cdot \|u_{(i,l)}\|} \quad (16)$$

where  $s, u_{(i,l)}$  denote a joint embedding vector and an embedding of user  $l$  who is neighbor of the target user  $i$ . Afterward, a fully connected layer is used to create the attention score and normalize by applied a softmax function. To perform the final neighbors' vectors, the neighbor's embedding are multiplied with their attention vectors dependent on the target user. Next, the friend-level attentions ( $\beta_{(i,l)}$ ) are computed using a two-layer NN:

$$\beta_{(i,l)} = h^T \text{ReLU}(W_1 u_i + W_2 v_j + W_3 f_{(i,l)} + b) \quad (17)$$

where  $u_i, v_j$ , and  $f_{(i,l)}$  are the target user, the target item, and neighbor vector, respectively.  $W_1 \in R^{d \times k}$ ,  $W_2 \in R^{d \times k}$ ,  $W_3 \in R^{d \times k}$ ,  $b \in R^k$ , and  $h \in R^k$  are model parameters.  $k$  and ReLU are the attention network dimension and a nonlinear activation function, respectively. After the friend-level attention is obtained, the target user embedding is modified by (18) as follows:

$$U_i = u_i + \sum_{l \in S_i} \beta_{(i,l)} f_{(i,l)} \quad (18)$$

where  $U_i$  denotes the modified target user embedding. Finally, the prediction score is performed by using a matrix factorization concept between the modified target user embedding and the target item embedding. However, SAMN apply a neighbor's attention, which is the similarity between users, with the neighbor's embedding instead of neighbor's rating as the CF' concept.

Current NCFs with friends, CMN and SAMN, give weight to the target user's neighbor by fusing them with the neighbor embedding. However, a neighbor embedding is the neighbor profile representation in terms of a numeric vector. Combining the similarity value as a weight with the neighbor embedding makes the user representation change. Moreover, it will cause the users' representation to deviate from the real user profile and influence incorrect predictions. The similarity values should combine with the user's rating for computing the target user's predicted rating on the target item as in the CF's prediction equation (Equation 2).

Table 1 shows the comparison of the neural based CF's related works. "/" denote the presence of the neighbor concept in each column and "X" represents the absence of the neighbor concept. "-" implies that there is no further comparison as the related work does not apply the neighbor's concept. In summary, the NCF, ONCF, and NNCF utilized the neural network with the collaborative filtering technique but did not apply the neighbor's concept to perform the predicted rating. BPAM, CMN, and SAMN proposed adopting the neighbors and performing the similarity between the neighbors and the target user differently. However, they did not build and implement the neighbors' ratings as the CF's prediction equation in the prediction process. Moreover, no research converts the neighbor's preference range into the target user preference aspect. The proposed method transforms the neighbor's preference range into the target user aspect. It also applies the neighbor's concept by utilizing the similarity value with the neighbor's rating.



*Table 1 Comparison of the related works with respect to the neighbor concept*

<b>Methods</b>	<b>Using neighbors</b>	<b>Calculating similarity</b>	<b>Applying the neighbors' ratings in the prediction process</b>	<b>Converting preference range</b>
<b>NCF</b>	X	-	-	-
<b>ONCF</b>	X	-	-	-
<b>NNCF</b>	/	X	X	X
<b>BPAM</b>	/	/	X	X
<b>CMN</b>	/	/	X	X
<b>SAMN</b>	/	/	X	X
<b>Proposed Method</b>	/	/	/	/

## 2.4 RATING CONVERSION

In the collaborative filtering concept, the neighbors' rating is used to perform the target user's predicted rating directly. However, these neighbors' ratings and the target user rating are in different ranges. For example, suppose that the rating range is 1-5. User A gives ratings in the 1-3 range for “dislike” to “like,” while user B gives ratings in the 3-5 range instead. This means that “like” of A (rating 3 of A) equals “dislike” of B (rating 3 of B). Thus, using one user's rating to predict another user's rating directly may cause incorrect prediction. In order to consider the neighbors and the target user preferences, the preference range must be considered in the same range. Thus, the method that converts the neighbor's rating pattern into the target user's rating pattern is called rating conversion.

To solve the rating conversion issue, some researchers applied the normalization technique, which maps all users' ratings into a range from 0 to 1. However, it is still not effective enough. Suppose two target users have the same group of neighbors. Both target users will receive the same recommendation or predicted rating. For example, target users  $u_1$  and  $u_2$  usually have normalization rates of 0.4 and 0.7, respectively. The predicted ratings of both the target users on the target item are 0.8. Therefore, this target item should be recommended to user  $u_1$  rather than user  $u_2$ . Because the predicted rating of the target item is in the range that user  $u_1$  likes, while it is in the average range of user  $u_2$ . W's transpose function [25] is proposed to solve the rating

conversion issue. This work transposes the user's rating into the target user's aspect using a relation between users' and the target user's rating pattern. There are four key terms in the  $W$ 's transpose function: original value, adjustment, confidence, and distribution. The original value means the rating that will be transposed. The adjustment means the average difference of co-rated items between the user and the target user. The confidence term ( $Conf_A$ ) and distribution term ( $Dist_{Au}$ ) are added to obtain more accurate results. The  $W$ 's transpose function ( $W_{u \rightarrow A}(s)$ ) is defined as follows:

$$Dist_{Au} = \frac{1}{\sigma(\{r_{Ai} | i \in \beta_{us}\}) + 1} \quad (19)$$

$$Conf_A = \frac{1}{\frac{\sum_{i \in I_A} |r_{Ai} - \hat{r}_{Ai}|}{|I_A|} + 1} \quad (20)$$

$$W_{u \rightarrow A}(s) = s + \frac{\sum_{i \in \beta_{us}} (r_{Ai} - s)}{|\beta_{us}|} \quad (21)$$

where  $r_{Ai}$  denotes the actual rating of the target user  $A$  on item  $i$ ,  $\hat{r}_{Ai}$  is the latent rating which is predicted from the latent model,  $\beta_{us}$  is the set of items that the target user  $u$  has rated score  $s$ ,  $I_A$  is the set of items that the target user  $A$  has rated.

## 2.5 REGION EMBEDDING

Recently, the region embedding method [26] is a new embedding technique that uses the words in the same region to perform the word representation. The region embedding assumption is that one embedded word in a different region of the document should not be the same representation. A representation of the continuous subsequence of the words in the document is called the text region. Moreover, the region embedding can solve the sparsity problem of the  $n$ -grams method. Suppose the sentence is "The story is sweet and simple and easy to read." The text region with length five means "story is sweet and simple." There are three steps to the region embedding method, as shown in Fig.1, to obtain the region embedding vector.

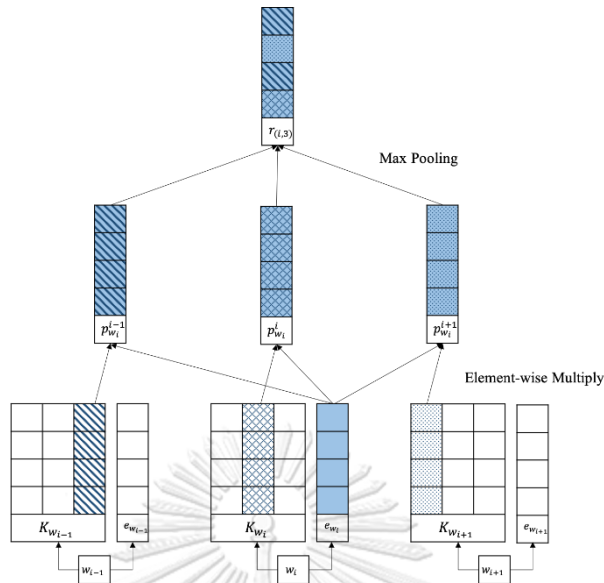


Figure 3 Context-Word Region embedding Method

First, input words in the text region are selected to produce the target word embedding vector and local context unit (LCU), a word's weight matrix. Both the target word embedding vector and LCU are produced applying the embedding method. Second, the LCU is projected into the target word embedding vector by utilizing the element-wise multiplication operation:

$$p_{w_{i+t}}^i = K_{w_{i,t}} \odot e_{w_{i,t}}, \quad (22)$$

where  $p_{w_{i+t}}^i$ ,  $K_{w_{i,t}}$ ,  $e_{w_{i,t}}$  denote the projected word embedding of  $w_{i+t}$  in  $i^{th}$  word, local context unit, and the embedding of word  $w_{i+t}$ . Afterward, the projected word embeddings from the previous step are combined using max-pooling to perform the word representation in this text region.

## CHAPTER 3 PROPOSED METHOD

In this chapter, the overview, process, and time complexity of the proposed are described. The overview of the proposed method illustrates the two issues that need to be considered when applying a neural network with the collaborative filtering concept. The process of the proposed method explains steps and illustrates the architecture of the proposed method, which can solve the two issues. The time complexity clarifies step-by-step to calculate the neural network time complexity of the proposed method.

### 3.1 OVERVIEW OF PROPOSED METHOD

In region embedding, the word in the same region as the target word is called the LCU. For example, suppose the target word “Apple” is in the two regions with the LCU1 and LCU2 for region1 and region 2, the word “Apple” is represented as “Apple 1 ” and “Apple 2,” respectively . Due to the difference between the relation of the target word and each LCU. The representation of the target word is different in each region.

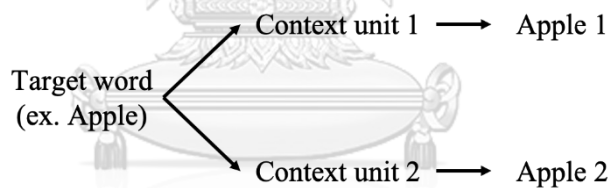


Figure 4 Context unit example

The target user is viewed as a target word when utilizing the region embedding assumption with the CF concept. The target user and neighbors are viewed as the LCU and the target word, respectively. Therefore, the target user will provide the different predicted ratings for the two different groups of neighbors, similar to the “Apple 1” and “Apple 2” in the region embedding example Figure 4.

In CF, the neighbor's concept is applied to the model in the prediction step. The neighbors' rating scores and the similarity levels between the neighbors and the target user are integrated to predict the target user's rating score as in Equation 2. In the past, no NCF research employed neighbors' ratings in the prediction process as in the traditional CF technique. When applying a

neural network with a collaborative filtering concept, two issues need to be considered: the similarities between the neighbors and the target user and the rating conversion. The assumption of region embedding is used to solve the rating conversion issue, converting the neighbors' ratings range into the target user aspect. The similarity between the neighbor and the target user issue is solved by capturing neighbors' attentions to perform the similarity level between the neighbors and the target user.

In the case of similarity issue, the neighbors' concept is the CF technique's key idea. It uses the neighbors' preferences to create the similarity levels between the neighbors and the target user and apply the neighbor's rating to perform the predicted rating. Many techniques are utilized to perform the similarity values between the neighbor and the target user, such as cosine similarity and Pearson's correlation. The similarity levels between the neighbors and the target user are employed to weigh the neighbors' ratings to compute the target users' ratings. For example, a rating range of 1-5 for "dislike" to "like" and a similarity range of [0,1] for "not similar" to "similar." The target user has two neighbors  $N_1$  and  $N_2$  in Figure 5. The similarity values between the target user and each neighbor  $N_1$  and  $N_2$  are 1 and 0.5, respectively. If both neighbors gave a score of 5 to the target item, but the similarity value is different. The ratings from  $N_1$  and  $N_2$  are transferred to the target user depending on the similarity level of each neighbor. Therefore, the target user's rating predictions via  $N_1$  and  $N_2$  are 5 and 2.5, respectively.

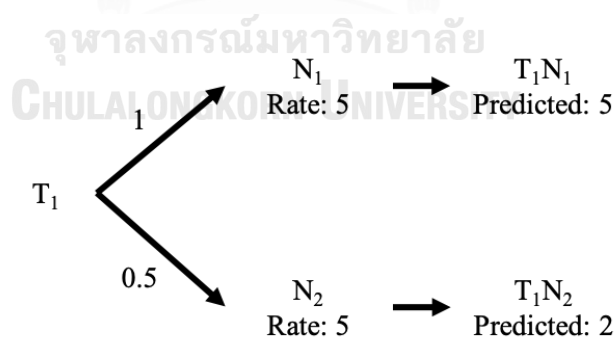


Figure 5 Similarity among users example

In the rating conversion issue, because of different users having different rating ranges, the neighbors' ratings are mapped from the neighbors' rating range to the target user's rating range. For example, the target user gives ratings in the range of 1-5 for "dislike" to "like," while  $N_1$  gives ratings in the range 1 to 3 and  $N_2$  gives rating in range 3 to 5 instead.

In the case of the rating conversion issue, the different users have different rating ranges. However, utilizing the neighbor's rating should be considered in the same range. Therefore, the neighbors' ratings are mapped from the neighbors' rating range to the target user's rating range. Suppose the target user gives ratings in the range of 1-5 for "dislike" to "like," while  $N_1$  gives ratings in the range 1 to 3 and  $N_2$  gives rating in range 3 to 5 instead.

Both neighbors gave 3 scores to the target item, as shown in Figure 6. If the rating conversion is not considered, scores of 3 from the neighbors will transfer directly to the target user, which means neutral for the target user. However, neighbor  $N_1$  wants to tell the target user to like the target item, which is a 5 score for the target user. Neighbor  $N_2$  wants to tell the target user not to like this item, which is a 3 score for the target user. Therefore, the rating conversion should be considered in the model to transfer the real neighbor's preference.



Figure 6 Rating conversion issue example

In order to solve both issues, three modules are proposed: the rating conversion module, the similarity module, and the prediction module. The rating conversion module projects the neighbor's characteristics with each target user's perspective. In other words, the neighbors will be converted into the term of target user aspect, which is similar to the region embedding local context unit's concept. The similarity module captures the neighbors' attentions to create the similarity levels between the neighbors and the target user. The prediction module combines the rating conversion and the similarity module result to imitate the collaborative filtering's prediction equation.

### 3.2 PROCESS OF PROPOSED METHOD

The proposed method process consists of five steps to perform the predicted rating: selecting neighbors, user-item embedding, converting neighbors into the target user aspect, crating similarity, and rating prediction. The selecting neighbor describes how to select the neighbor of the target user. The user-item embedding step explains the process of creating the user-item representation. The converting neighbors into the target user aspect and crating similarity steps show the solution process to solve the two issues when applying a neural network with the CF concept. The rating prediction step integrates the converted neighbors' ratings and the similarity value to make predictions that imitate the CF prediction equation. The architecture of the proposed model is shown in Figure 7.

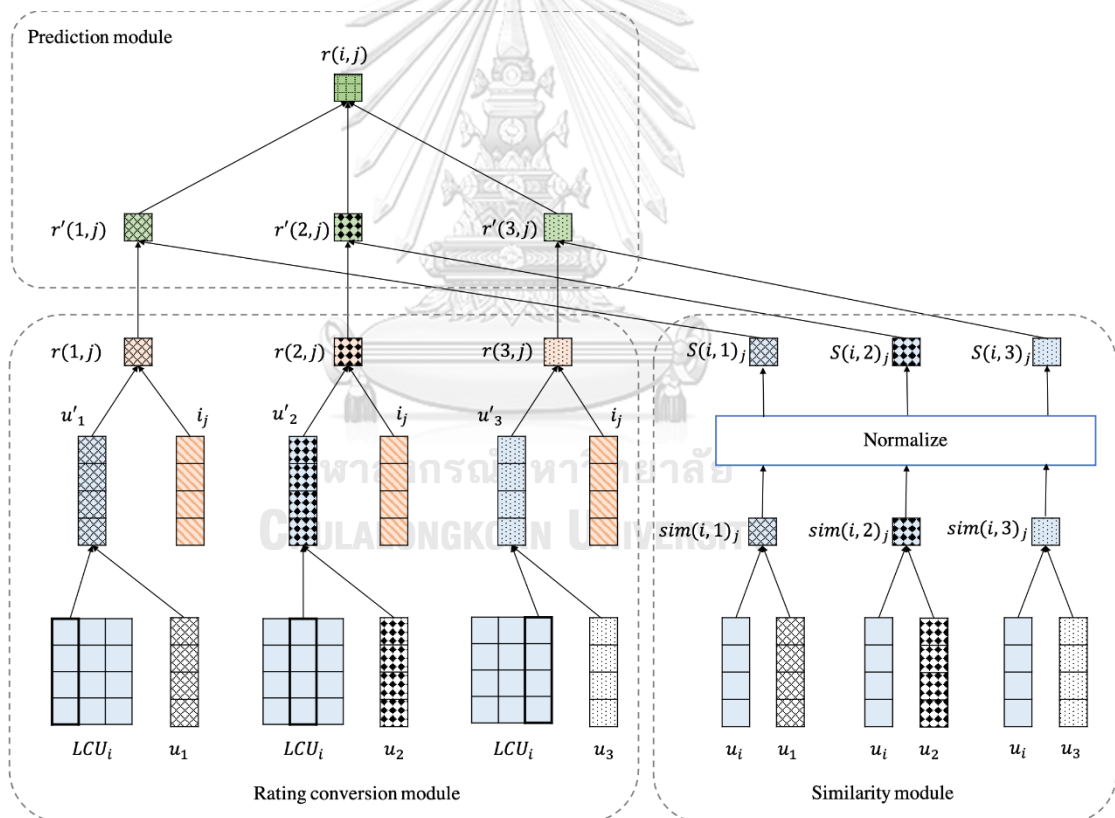


Figure 7 The proposed model architecture

### 3.2.1 SELECTING NEIGHBORS

Applying the neighbors' concept to predict the target user's rating on the target item is crucial in the CF technique. In order to select neighbors of each target user, the raters who rate the target item are chosen. Figure 8 shows three cases of raters: the number of raters less than  $N$ , equal to  $N$ , and more than  $N$ . The target item A and B are examples of a number of raters less than  $N$  and equal to  $N$ , respectively. In these two cases of raters, all raters are selected as neighbors. While, Target item C is an example of a number of raters greater than  $N$ . In this case, the raters are randomly selected as the target user's neighbors. Therefore, the outcome of this step is a list of  $N$  neighbors or lower than  $N$  neighbors. Figure 8 shows the three example case of selecting neighbor.

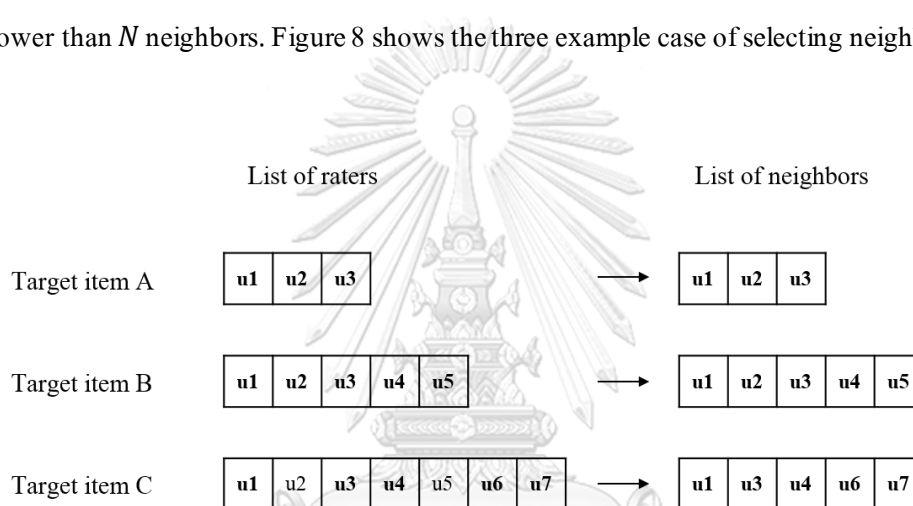


Figure 8 The example of neighbors selection

### 3.2.2 USER-ITEM EMBEDDING

After preparing the neighbors' list, the proposed method inputs are the target user, the target item, and the list of neighbors. In each user's interaction, the user and item are initialized in the equal embedding size to represent the target user and target item. The neighbors' list is initialized as the neighbors' matrix, and each row is the neighbor's embedding. In order to learn the neural network, the weights of the neighbors are also initialized separately. If the number of the rater is less than  $N$ , the neural network weights of all rater are initialized. The remaining neighbors' weights are assigned zero. For example,  $N$  is set to ten. Therefore, ten neighbors' embeddings are initialized. Suppose the number of raters is six, which is less than ten, then the model's weights of six neighbors are initialized. The remaining four weights are assigned as zero, which is these four neighbors' embedding will not be learned. In training a neural network, the target user vectors, the



target item vectors, and neighbors' matrix will be adjusted by using the actual ratings of the target user as a label of the model.

### 3.2.3 CONVERTING NEIGHBOR INTO THE TARGET USER ASPECT

In order to solve the rating conversion issues, the region embedding LCU concept [26] is applied. In region embedding, the LCU represents each word's influence in the region on the target word, which is used to perform the target word representation in that region. In the same way, the LCU is applied with the neighbor concept to create the neighbor representation in the target user's perspective view. The rating conversion module in Figure 7 shows converting the neighbor's preferences into the target user aspect. First, the target user  $i$  LCU ( $LCU_i$ ) is initialized in the size-by-number of the neighbors' embedding matrix. In Figure 7, suppose there are three neighbors and the embedding size is set to four. Therefore, the target user's LCU dimension is a four-by-three matrix. Afterward, the target user's LCU of  $n$  th neighbor ( $LCU_{(i,n)}$ ) is intergraded with the neighbors' embeddings ( $u_n$ ) using the element-wise multiplication. Therefore, the projected neighbors' representations in the target user perspective ( $u'_n$ ) are obtained.

$$u'_n = LCU_{(i,n)} \odot u_n \quad (23)$$

To compute the neighbors' rating scores in the target user aspect ( $r_{(n,j)}$ ), these projected neighbors' representations and the target item's embedding ( $i_j$ ) are combined using matrix factorization concept (Equation 24).

$$r_{(n,j)} = u'_n \cdot i_j \quad (24)$$

In order to adjust the target user's LCU, the neighbors' embeddings, and other parameters in the proposed model, the actual ratings of the target user are labeled and learned by applying the backpropagation algorithm. The outcome of this step is the neighbors' rating in the target user aspect, which represent the neighbors' preferences in the target user perspective view.

### 3.2.4 CREATING SIMILARITY

Usually, the similarity levels between the neighbors and the target user are calculated using cosine similarity or Pearson's correlation equation on a pair of the target user and each neighbor. The similarity levels between the neighbor and target user in this work are the attention between the neighbor and the target user. In order to create the attention among users, the target user vector ( $u_i$ ) and the neighbor vector ( $u'_n$ ) are integrated by using a dot product:

$$sim(i, n)_j = u_i \cdot u'_n \quad (25)$$

After that, the softmax function (Equation 25) is applied to adjust the similarity value into the range  $[-1, 1]$ . In order to select the high-quality neighbors, the neighbors who have a similarity value more than zero are used to perform the predicted rating in the prediction module. Therefore, the similarity range of the target user is in  $[0, 1]$ , which is the same range as the similarity value range of the CF technique. The creating similarity step's architecture is shown in the similarity module in Figure 7.

$$S(i, n)_j = \frac{\exp(sim(i, n)_j)}{\sum_{n \in N} \exp(sim(i, n)_j)} \quad (26)$$

### 3.2.5 RATING PREDICTION

In the collaborative filtering technique, the neighbor's ratings are integrated into the prediction process to predict the target user's rating. The same as the CF approach, the rating prediction equation (Equation 2) is imitated into the proposed method in the rating prediction step. In this step, the neighbor's rating in the target user aspect and the similarities between the target user and their neighbors are combined. Due to different neighbors having a different similarity level toward the target user, the neighbors' rating ( $r(n, j)$ ) needs to be weighed. Therefore, the predicted rating scores of the target user are performed by using a weighted average on the neighbors' ratings where the similarity levels are utilized as a weight. The proposed method outcome is the predicted rating of the target user  $i$  on the target item  $j$  ( $r(i, j)$ ). The prediction module in Figure 7 is the architecture of the rating prediction step.

$$r'(n, j) = r(n, j) \cdot S(i, n)_j, \quad (27)$$

$$r(i, j) = \frac{\sum_{n \in N} r'(n, j)}{\sum_{n \in N} S(i, n)_j} \quad (28)$$

The predicted ratings of the target user are used to compare with the actual ratings of the target user to compute the prediction errors. These errors are used to update the weight of the neural network and the model parameters by using backpropagation.

### 3.3 TIME COMPLEXITY OF PROPOSED METHOD

The time complexity for one training epoch a neural network is calculated. Due to the prediction module is not a neural network layer. Therefore this module is not used to calculate the time complexity of the proposed method. The rating conversion module use three layers to perform the neighbor's rating in the target user aspect ( $r'(1, j)$ ), which are input layer, hidden layer, and output layer. The similarity module is a sub neural network which use two layer: input layer and output layer. The neural network architecture of the rating conversion module and the similarity module are shown in Figure 9.

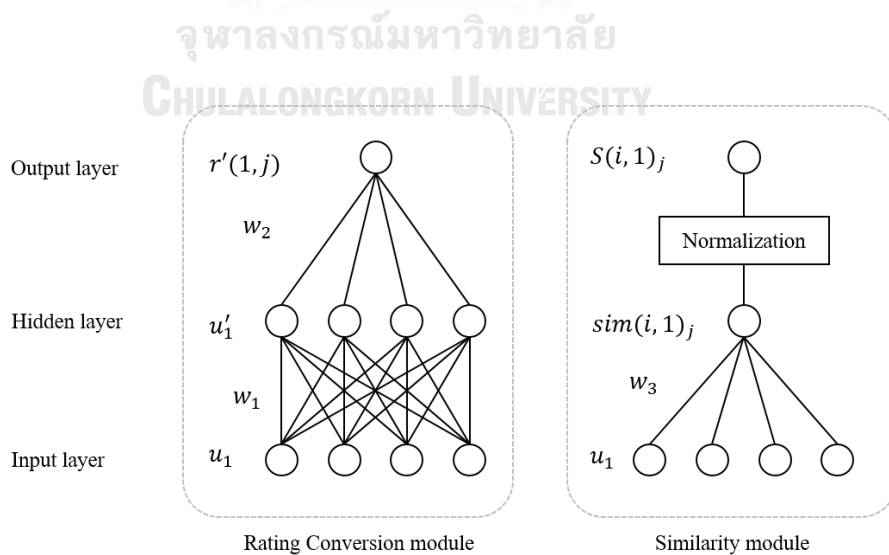


Figure 9 The proposed method neural network architecture

The input of the rating conversion module is the neighbor embedding size ( $e_n$ ). Suppose neighbor  $u_1$  uses four embedding sizes, four nodes, to represent the user profile. To convert the neighbor representation into the target user aspect ( $u'_1$ ), the neighbor embedding is multiplied with the target user's LCU ( $w_1$ ) using element-wise multiplication. Therefore, the converted neighbor representation embedding size ( $e_{cn}$ ) is equal to the neighbor embedding size. In order to perform the neighbor's rating in the target user aspect, the neighbor representation in the target user aspect is multiplied with the item embedding ( $w_2$ ). Thus, feedforward time complexity of the rating conversion module calculate as follow.

Assume there are  $t$  neighbors. For propagating from layer  $u_1$  to layer  $u'_1$ .

$$U_{u'_1 t} = W_{u_1 u'_1} \odot Z_{u_1 t} \quad (29)$$

Then, apply the activation function

$$Z_{u_1 t} = f(U_{u_1 t}) \quad (30)$$

This has  $O(e_{cn} t)$  the time complexity because it is an element-wise operation.

In total, layer  $u_1$  to layer  $u'_1$  the time complexity is

$$O(e_{cn} e_n t + e_{cn} t) = O(e_{cn} t * (e_n + 1)) = O(e_{cn} e_n t) \quad (31)$$

Therefore, layer  $u'_1$  to  $r'(1, j)$ , the time complexity is  $O(e_r e_{cn} t)$  where  $e_r$  is the number of  $r'(1, j)$  nodes.

Thus, the total time complexity for feedforward propagation will be

$$O(e_{cn} e_n t + e_r e_{cn} t) = O(e_{cn} t * (e_n + e_r)) \quad (32)$$

Because the output layer is the rating prediction score, which is one node. Therefore, the total time complexity for feedforward propagation of the rating conversion module is  $O(e_{cn} e_n t)$ .

In order to calculate the backpropagation time complexity, we calculate from the output layer to the input layer. Firstly, we compute the error  $E_{e,t}$ , a matrix contain the error for nodes at layer  $r'(1, j)$ .

$$E_{r'(1,j)t} = f'(U_{r'(1,j)t}) \odot (Z_{r'(1,j)t} - O_{r'(1,j)t}) \quad (33)$$

where  $\odot$  means element-wise multiplication.  $E_{r'(1,j)t}$  has  $r$  rows and  $t$  columns, which means each column is the error signal for neighbor  $t$ .  $Z_{r'(1,j)t}$  and  $O_{r'(1,j)t}$  are the predicted rating and the actual rating, respectively.  $f'(U_{r'(1,j)t})$  is the differential of the activation function.

Then, compute delta weight between layer  $u'_1$  and  $r'(1, j)$  ( $D_{r'(1,j) u'_1}$ )

$$D_{e_r e_{cn}} = E_{e,t} * Z_{te_{cn}} ; Z_{te_{cn}} \text{ is the transpose of } Z_{e_{cn} t} \quad (34)$$

Therefore, the weight between layer  $u'_1$  and  $r'(1, j)$  ( $W_{r'(1,j) u'_1}$ ) are adjusted.

$$W_{r'(1,j) u'_1} = W_{r'(1,j) u'_1} - D_{r'(1,j) u'_1} \quad (35)$$

Therefore, the time complexity of layer  $u'_1$  to  $r'(1, j)$  is

$$\begin{aligned} O(e_r t + e_r t + e_r t e_{cn} + e_r e_{cn}) &= O(2e_r t + e_r e_{cn}(t+1)) \\ &= O(2e_r t + e_r e_{cn} t) \\ &= O(e_r e_{cn} t) \end{aligned} \quad (36)$$

The backpropagation of layer  $u'_1$  to  $r'(1, j)$ ,

$$E_{u'_1 t} = f'(S_{w_1 t}) \odot (Z_{u'_1 r'(1,j)} - E_{r'(1,j)t}) \quad (37)$$

$$D_{u'_1 u_1} = E_{u'_1 t} * Z_{tu_1} \quad (38)$$

$$W_{u'_1 u_1} = W_{u'_1 u_1} - D_{u'_1 u_1} \quad (39)$$

Therefor the time complexity of layer  $u'_1$  to  $u_1$  is

$$\begin{aligned}
 O(e_{cn}t + e_{cn}e_r t + e_{cn}te_n + e_{cn}e_n) &= O(e_{cn} * (t + e_r t + te_n + e_n)) \\
 &= O(e_{cn} * (t(1 + e_r) + e_n(t + 1))) \\
 &= O(e_{cn} * (te_r + e_n t)) \\
 &= O(e_{cn} * t(e_r + e_n))
 \end{aligned} \tag{40}$$

The total time complexity of the rating conversion module backpropagation is

$$\begin{aligned}
 O(e_r e_{cn} t + e_{cn} t(e_r + e_n)) &= O(e_{cn} t(e_r + (e_r + e_n))) \\
 &= O(e_{cn} t(2e_r + e_n)) \\
 &= O(e_{cn} te_n)
 \end{aligned} \tag{41}$$

When considering the time complexity of the similarity module, this module's input is the neighbor embedding size similar to the rating conversion module. From the example Figure 9, neighbor  $u_1$  uses four embedding sizes, four nodes. In order to perform the neighbor attention, the neighbor's embedding is multiplied by using dot product with the target user embedding ( $w_3$ ), which is viewed as the weight of the neural network. After the similarity between the neighbor and the target user ( $sim(i, 1)_j$ ) is created, the activation function is applied. Therefore, the time complexity of the similarity module on  $t$  neighbors is  $O(e_n t)$ .

From the time complexity of both modules, it is clear that the rating conversion module has a higher time complexity than the similarity module. Therefore, the time complexity of the proposed method depends on the rating conversion module. In summary, the time complexity of the rating conversion module feedforward is equal to the time complexity of the rating conversion module backpropagation, which are  $O(e_n e_{cn} t)$  for one epoch. In this work, the neighbor embedding and the neighbor in the target user aspect embedding are the same sizes. Due to training the model, the number of epochs in each dataset is different. Therefore, the time complexity of the proposed method depends on the neighbor embedding size, the number of neighbors, and the number of epochs.

## CHAPTER 4 EVALUATION

In the real world, users' interactions are collected in various formats such as ratings, reviews, and images. The form that is popularly employed for evaluating the CF model is the ratings of users on items. However, evaluating on real-world datasets is unable to control the data distribution. Therefore, the proposed method is assessed on both real-world and synthetic datasets. The proposed method is compared with the existing NCF with friends methods, CMN [21] and SAMN [22]. Furthermore, I want to know the proposed model can overcome the latent factor models or not. The Singular Value Decomposition (SVD) and Non-negative matrix factorization (NMF) are the latent factor models that are used to compare with the proposed method. In CF research, the target user's neighbors are important. Therefore, two types of utilizing neighbors are evaluated in this experiment to analyze the performance of using a different number of neighbors:  $N$  neighbors and all neighbors. Moreover, the effectiveness of the rating conversion module of the proposed method is also assessed by removing the target user's LCU from the rating conversion module. In other words, the neighbors' representations without the rating conversion are employed to create the neighbors' ratings directly via implementing the matrix factorization concept.

For the synthetic datasets, the proposed method is also compared with the SVD, NMF, CMN, and SAMN. In order to simulate the datasets, the full rating matrices and the partial rating matrix are constructed and compared with each other. The full rating matrixes are invented based on different distributions and dataset sizes. The partial rating matrix is a rating matrix that is comparable to the real world but in the normal rating distribution.

Due to GMF and NCF being two famous baselines, both CMN and SAMN have been demonstrated to outperform GMF and NCF. Thus, GMF and NCF will not be compared in this experiment. The ranking-based evaluation and prediction accuracy are applied to compare the ranking performance and accuracy of the proposed method.

### 4.1 REAL-WORLD DATASETS

In this subsection, the proposed method is evaluated by using both ranking-based evaluation and prediction accuracy on a real-world datasets. The three different categories of datasets are selected from the real-world dataset to determine whether the proposed method can be applied to different categories: movies, online products, and restaurants.

#### 4.1.1 DATA PREPARATION

To evaluate the performance of the proposed method, three public datasets are used:

- **MovieLens 1M (ML-1M):** This dataset is collected by grouplens. It is a consistent benchmark dataset with one million movie ratings and is commonly utilized in research experiments. The MovieLens 1M dataset consists of four attributes: the user ID, movie ID, users' ratings, and timestamp. Moreover, Each user in this dataset has at least twenty interactions.
- **Epinions:** This dataset is a who-trust-whom online social network that includes product ratings and reviews. It contains three attributes, which are the product name, category, and timestamp.
- **Yelp:** The businesses and services dataset in four countries contains users' data and reviews, ratings, pictures, and details. It is provided for the academic challenge and teaches students about databases.

Table 2 shows the number of interactions, users, and items in each dataset before preprocessing of three real-world datasets.

*Table 2 The number of records, users, and item in each dataset*

<b>Dataset</b>	<b>Interactions</b>	<b>Users</b>	<b>Items</b>
<b>MovieLens 1M</b>	1,000,209	6,040	3,706
<b>Epinions</b>	664,824	49,290	139,738
<b>Yelp</b>	8,021,122	1,968,703	209,393

In order to ensure that each user has enough interactions to be represented as a vector, users with more than twenty records are selected in the preprocessing step. Table 3 shows the remaining number of interactions, users, and items after preprocessing of three real-world datasets. The MovieLens dataset provides at least twenty records per user. Thus, the number of interactions, users, and items of the MovieLens 1M dataset in Table 3 is the same as in Table 2.



Table 3 The number of records, users, and item in each dataset after preprocessing

Dataset	Interactions	Users	Items
MovieLens 1M	1,000,209	6,040	3,706
Epinions	564,709	8,217	106,242
Yelp	2,253,312	57,814	31,943

#### 4.1.2 EXPERIMENTAL SETTINGS

In order to evaluate the dataset, each dataset is split in the ratio 80:10:10 for training:validation:testing sets. The parameters of the baseline method are set as in the corresponding papers. The proposed method's batch size is examined at {32, 64, 128}, and the learning rate is tested in {0.01, 0.05, 0.1}. In this work, I tried to use an embedding size at {8, 16, 32, 64} and neighbor size at {10, 20, 30, 40, 50}. In the ranking quality evaluation, the top  $k$  items for ranking is fixed, {5, 10, 20, 30, 40, 50}.

This work proposes to apply a neural network into the collaborative filtering concept, which considers similarity levels between the neighbors and the target user and the rating conversion. In this work, the adaptive momentum (Adam) [27], a combination of root mean square propagation gradient descent (RMSprop) [28] and stochastic gradient descent (SGD) [29], is utilized as an estimation optimizer to optimize the model. It is widely used in current works and requires small memory. The Adam optimizer calculated adaptive learning rates and the exponentially decaying average of previous gradients in the learning process for each parameter. The loss function in a multi-class classification task, called categorical cross-entropy, is employed to minimize the prediction error and learn the model parameters:

$$L = - \sum_{c=1}^C r_{(i,j)} \cdot \log y_{(i,j)} \quad (42)$$

where  $C$  denotes the number of output classes, which is a number of rating classes. In this work,  $C$  is set at 10 or 5 depending on the rating range in each dataset.  $r_{(i,j)}$  and  $y_{(i,j)}$  are the actual rating and predicted rating of the target user  $i$  on the target item  $j$ .

### 4.1.3 EVALUATION METRICS

There are two types of evaluation metrics to evaluate the proposed method: ranking-based evaluation and prediction accuracy. Two ranking quality metrics are used in this work: normalized discounted cumulative gain (nDCG) and hit ratio (HR). In order to indicate the prediction accuracy, there are three evaluation metrics: precision, recall, and RMSE.

#### 4.1.3.1 RANKING-BASED EVALUATION

The nDCG is one of the most popular metrics that compare the ranking quality using a relevant score. It is the recommended order ratio of Discounted Cumulative Gain (*DCG*) and the ideal order of DCG is called Ideal Discounted Cumulative Gain (*IDCG*). The nDCG equation defined as follows:

$$nDCG_p = \frac{DCG}{IDCG} \quad (43)$$

$$DCG_p = \sum_i^p \frac{2^{rel_i} - 1}{\log_2(i + 1)} \quad (44)$$

$$IDCG = \sum_{i=1}^{|REL_p|} \frac{2^{rel_i} - 1}{\log_2(i + 1)} \quad (45)$$

where  $p$ ,  $rel_i$ ,  $|REL_p|$  denote a particular rank position, the relevance of recommendation at position  $i$ , and the list of items ordered by relevance in the corpus up to position  $p$ .

The HR indicates the accuracy of top  $k$  in the recommendation list. If the item in the top  $k$  recommendation list matches the top  $k$  rated items list of the target user, it means “hit.” The number of hits divided by the number of all item in the test set is the HR value.

$$HR = \frac{\text{number of hits}}{\text{total number of items}} \quad (46)$$

#### 4.1.3.2 PREDICTION ACCURACY

Precision, recall, and RMSE metrics are used to evaluate the prediction accuracy. Precision measure the prediction exactness, the proportion of the recommended items in the top k set that are relevant. The precision metric is the number of the recommended items that are relevant divided by the total number of recommended items. The precision is defined as follows:

$$Precision = \frac{\text{relevant item}}{\text{recommended item}} \quad (47)$$

The recall is the relevant item proportion on a total number of relevant items, which is the number of the recommended items that are relevant divided by the total number of relevant items.

$$Recall = \frac{\text{relevant item}}{\text{all relevant item}} \quad (48)$$

Another way to evaluate the rating prediction accuracy is by measuring the error of the predicted rating. Therefore, a Root Mean Square Error (RMSE) is selected to evaluate the error of the proposed method. In order to evaluate by employing the RMSE, the predicted ratings that are higher than three scores are defined as relevant in this work. To calculate the prediction error, the predicted rating ( $r(i, j)$ ) is compared with the actual rating ( $\hat{r}(i, j)$ ). The RMSE is computed by averaging the sum of the error squared values over all predictions:

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^n (r(i, j) - \hat{r}(i, j))^2} \quad (49)$$

where  $N$  is the total number of the prediction.

#### 4.1.4 NEIGHBOR SELECTION

In the traditional CF technique, the similarity between the neighbor and the target user is integrated as weight on the neighbor's ratings. Thus, the neighbors of the target user are necessary for the CF research. In the proposed method, the neighbors are selected by two rules: the users who have rated the target item (raters) and the rater's similarity value more than zero. In creating the

similarity step, the attentions between the neighbor and the target user are normalized utilizing the softmax function to create the similarity value between the neighbor and the target user. The similarity between the neighbor and the target user in the traditional CF technique is in the range  $[0,1]$ . In this work, the raters who have similarity more than zero are chosen as the neighbors. Therefore, the similarity between the neighbor and the target user is in the range  $[0,1]$  as in the CF technique. This similarity value means how similar the neighbor is to the target user. Suppose the similarity value of the neighbor close to one means this neighbor is most similar to the target user. In contrast, if the similarity value is close to zero, this neighbor is a low-quality neighbor. Due to a large number of raters in each dataset, using all raters as the neighbors can make the incorrect prediction because some neighbors are the noise of recommendation. In order to find the effectiveness of the number of neighbors, this work divided using neighbors into two types:  $N$  neighbors and all neighbors. The  $N$  neighbors are the randomly selected  $N$  raters as neighbors. In contrast, all neighbors use all rater to be the neighbors.

#### 4.1.4.1 $N$ NEIGHBORS EXPERIMENTAL RESULTS

The proposed method aims to solve the two issues: the similarities between the neighbors and the target user and the rating conversion. The neighbors' concept is considered and applied to the neural network model. The  $N$  neighbors are employed to predict the preference of the target user. Because the similarity issues are directly associated with the number of neighbors. It is an essential hyperparameter that needs to be tuned. The embedding size of the target user, neighbor, and item representations are other essential hyperparameters used to represent the user-item characteristics via ratings. The MovieLens 1M dataset's experimental results are shown in Figure 10. This figure presents the experimental results when using different embedding sizes and a different number of neighbors.

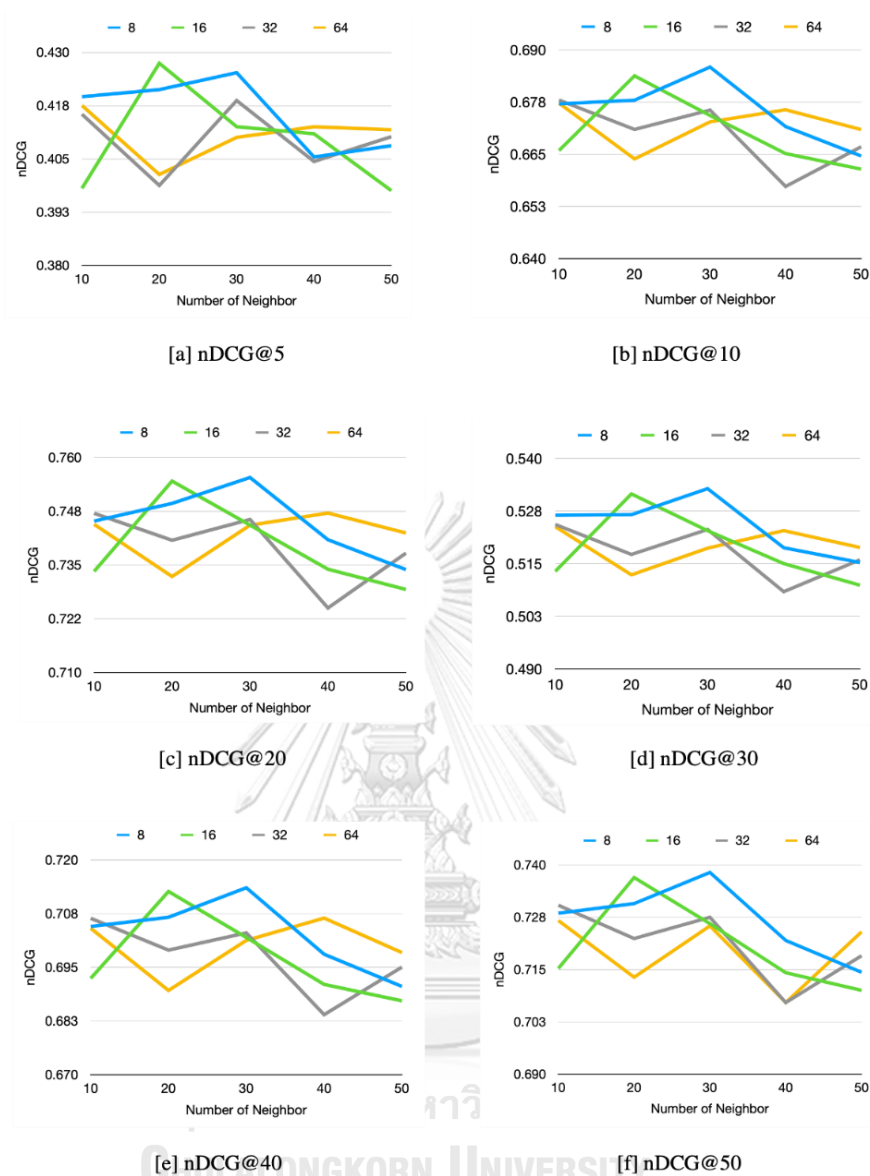


Figure 10 Experimental results of MovieLens 1M dataset for the proposed method varying the embedding size and neighbors.

Figure 10 [a-f] shows the experimental results of embedding size of 8, which obtains the highest nDCG results when the numbers of neighbors were 10 and 30 compared with the embedding sizes of 16, 32, and 64. Even though the embedding size of 16 provided the highest performance when using 20 neighbors in Figure 10 [a], using 10, 30, 40, and 50 neighbors gave lower nDCG results than the other three embedding sizes. In the same case as the embedding size of 16, the embedding size of 64 received the highest nDCG results when the number of neighbors was 50 in Figure 10 [a-e]. However, 10, 20, 30, and 40 neighbors at the embedding size of 64 obtained lower

nDCG results than the other three embedding sizes. Although the embedding size of 32 obtains the same trends as the embedding size of 8, the embedding size of 32 provides a lower nDCG result than the embedding size of 8. The overall nDCG results obtained from embedding size 8 look better than the other three embedding sizes, especially at a neighbor size of 30. In terms of the neighbor parameter, most of the nDCG results increase when neighbors reach 30. Afterward, the results decrease. The result proves that increasing the number of neighbors can improve the rating prediction accuracy. According to the experiment results, the first 30 neighbors contain more high-quality neighbors, and the number of neighbors more than 30 neighbors are low-quality neighbors. If the number of neighbors is too high, the low-quality neighbors will decrease the nDCG. Therefore, an embedding size of 8 and neighbors 30 are set as the proposed method to compare with the CMN and SAMN other baselines.

After that, I experimented on the Epinions and Yelp datasets using embedding sizes of {8, 16, 32, 64} and the number of neighbors {10, 20, 30, 40, 50}. Furthermore, for both datasets, the number of neighbors is varied in the same way as in the MovieLens 1M dataset. Overall, the best nDCG ranking results come from the Epinions and Yelp datasets, with embedding sizes of 64 and 32, respectively. To compare with the proposed method, the Epinions dataset with an embedding size of 64 and the Yelp dataset with an embedding size of 32 were employed.

Table 4 Comparison of the experimental results on all datasets using nDCG

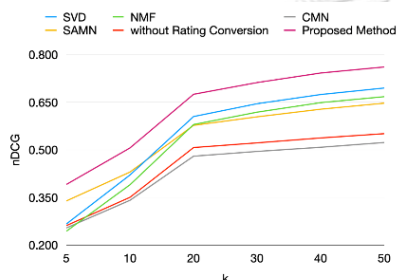
Datasets	Methods	nDCG@k					
		5	10	20	30	40	50
ML-1M	SVD	0.267	0.421	0.605	0.645	0.674	0.695
	NMF	0.244	0.390	0.580	0.619	0.649	0.667
	CMN	0.254	0.341	0.480	0.495	0.508	0.523
	SAMN	0.339	0.430	0.577	0.604	0.628	0.647
	Without Rating Conversion	0.351	0.350	0.507	0.522	0.537	0.551
	Proposed Method	<b>0.391</b>	<b>0.506</b>	<b>0.507</b>	<b>0.522</b>	<b>0.537</b>	<b>0.551</b>
Epinions	SVD	0.717	0.752	0.775	0.750	-	-
	NMF	0.700	0.747	0.767	0.761	-	-

	CMN	0.698	0.701	0.678	0.701	-	-
	SAMN	0.745	0.772	0.760	0.738	-	-
	<b>Without Rating Conversion</b>	0.720	0.717	0.722	0.733	-	-
	<b>Proposed Method</b>	<b>0.749</b>	<b>0.783</b>	<b>0.775</b>	<b>0.783</b>	-	-
Yelp	SVD	0.610	0.627	0.639	0.641	0.648	0.653
	NMF	0.592	0.613	0.629	0.632	0.634	0.636
	CMN	0.633	0.614	0.600	0.592	0.591	0.588
	SAMN	0.636	0.669	0.660	0.657	0.605	0.619
	<b>Without Rating Conversion</b>	0.654	0.643	0.633	0.628	0.628	0.627
	<b>Proposed Method</b>	<b>0.675</b>	<b>0.671</b>	<b>0.665</b>	<b>0.663</b>	<b>0.665</b>	<b>0.665</b>

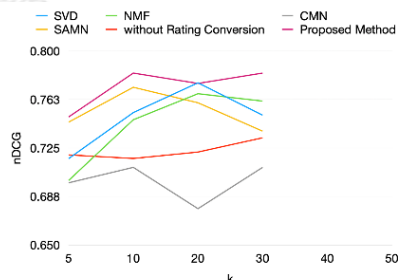
Table 5 Comparison of the experimental results on all datasets using the HR

Datasets	Methods	HR@k					
		5	10	20	30	40	50
ML-1M	SVD	0.259	0.415	0.635	0.687	0.729	0.758
	NMF	0.239	0.387	0.619	0.671	0.715	0.749
	CMN	0.286	0.401	0.598	0.637	0.669	0.697
	SAMN	0.325	0.440	0.627	0.674	0.709	0.736
	<b>Without Rating Conversion</b>	0.290	0.404	0.628	0.670	0.707	0.734
	<b>Proposed Method</b>	<b>0.337</b>	<b>0.455</b>	<b>0.654</b>	<b>0.695</b>	<b>0.733</b>	<b>0.761</b>
Epinions	SVD	0.771	0.829	0.905	0.970	-	-
	NMF	0.748	0.797	0.876	0.948	-	-
	CMN	0.760	0.810	0.881	0.962	-	-
	SAMN	0.802	0.843	0.923	0.980	-	-
	<b>Without Rating Conversion</b>	0.774	0.823	0.919	0.979	-	-

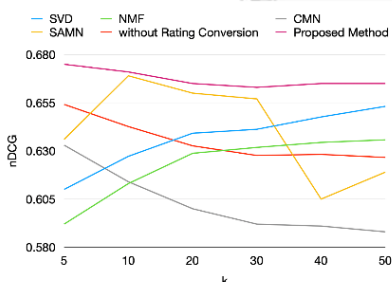
	<b>Proposed Method</b>	<b>0.808</b>	<b>0.860</b>	<b>0.939</b>	<b>0.981</b>	-	-
<b>Yelp</b>	<b>SVD</b>	0.678	0.714	0.744	0.756	0.769	0.777
	<b>NMF</b>	0.665	0.704	0.736	0.749	0.760	0.769
	<b>CMN</b>	0.709	0.729	0.754	0.764	0.773	0.780
	<b>SAMN</b>	0.585	0.732	0.778	0.799	0.802	0.830
	<b>Without Rating Conversion</b>	0.729	0.729	0.755	0.769	0.775	0.793
	<b>Proposed Method</b>	<b>0.792</b>	<b>0.788</b>	<b>0.785</b>	<b>0.803</b>	<b>0.806</b>	<b>0.837</b>



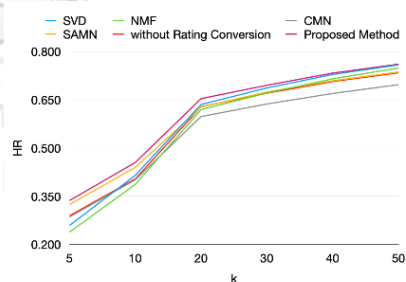
[a] nDCG ML-1M dataset



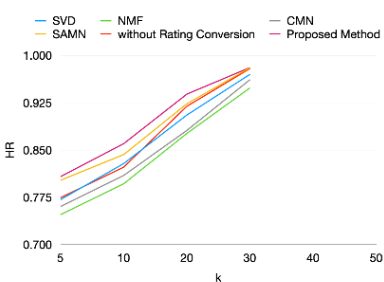
[b] nDCG Epinions dataset



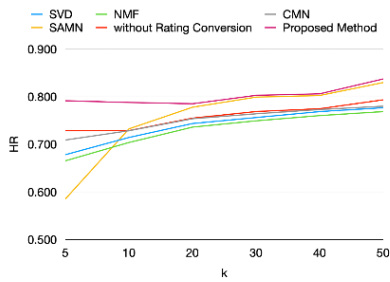
[c] nDCG Yelp dataset



[d] HR ML-1M dataset



[e] HR Epinions dataset



[f] HR Yelp dataset

Figure 11 Experimental results graph comparison on all datasets using the nDCG and HR



Table 4 and Figure 11 [a-c] illustrate the nDCG experimental results of the proposed method, latent factor models, and current NCFs with friends on the MovieLens 1M, Epinions, and Yelp datasets utilizing 30 neighbors at embedding sizes of 8, 64, and 32. The HR experimental results are shown in Table 5 and Figure 11 [d-f]. The proposed method obtains significantly higher results than SVD, NMF, CMN, and SAMN at all  $k$ . Although some of the proposed method outcomes are slightly lower than the SAMN method results, the proposed method takes significantly less time to process, approximately three hours less on the MovieLens 1M dataset. Since the Epinions dataset is a small dataset when compared with the other two datasets. Therefore, after preprocessing of this dataset, the remaining data does not have enough records to experiment on nDCG@40 and nDCG@50.

Table 6 Predicted accuracy experimental results

Datasets	Metrics	SVD	NMF	CMN	SAMN	Without Rating Conversion	Proposed Method
ML-1M	Precision	0.798	0.787	0.766	0.788	0.781	<b>0.811</b>
	Recall	0.782	0.773	0.224	0.776	0.766	<b>0.809</b>
	RMSE	0.988	1.025	1.282	1.029	1.262	<b>0.985</b>
Epinions	Precision	0.769	0.751	0.719	0.772	0.728	<b>0.789</b>
	Recall	0.775	0.746	0.749	0.798	0.785	<b>0.812</b>
	RMSE	1.413	1.474	1.637	1.410	1.580	<b>1.395</b>
Yelp	Precision	0.779	0.766	0.840	0.849	0.846	<b>0.866</b>
	Recall	0.785	0.749	0.768	0.776	0.773	<b>0.791</b>
	RMSE	1.112	1.230	1.232	1.076	1.127	<b>1.037</b>

Table 6 shows the experimental results of the proposed method versus other methods using prediction accuracy metrics on three datasets. When using precision and recall to compare all datasets, the proposed method performs much better than SVD, NMF, CMN, SAMN, and the proposed method without the rating conversion module. Moreover, the proposed method also obtains the lowest errors on the RMSE metric. Therefore, converting the neighbor's preference

range into the target user aspect and indicating the user similarity through the neighbors' attentions can provide more accurate experimental results.

The proposed method is compared to the proposed method without rating conversion to assess the effectiveness of the rating conversion module. The experimental outcomes are presented in Table 4-6 and Figure 11. Because the neighbors' preference ranges are converted into the target user aspect, the proposed method obtains higher efficiency than the proposed method without the rating conversion at all evaluation metrics. Additionally, the proposed method without the rating conversion can outperform CMN but is lower than the SAMN. As a result, converting the neighbors' ratings into the target user's perspective view can improve the prediction accuracy over not using the rating conversion.

When comparing the proposed method with latent factor models, the results are shown in Table 4-6 and Figure 11. The results show that the proposed method performs better than the SVD and NMF in the ranking-based evaluation and prediction accuracy. Therefore, the proposed method can overcome the latent factor model.

When comparing the proposed method without rating conversion experimental results with the other methods in terms of percentage, the results have shown in Table 7-9. Table 7 and 8 show the difference between the other methods and the proposed method without rating conversion using nDCG and HR, respectively. The green numbers refer to percentage of the proposed method without rating conversion outperform that method. From Table 9, the green numbers mean the proposed method without rating conversion obtained higher performance than that method using precision or recall metric. Due to the RMSE metric used to compare the error, the lower the error value, the better. Therefore, the blue numbers indicate that the proposed technique without rating conversion is better than the existing method. In summary, the proposed method without rating conversion, which directly combines neighbor's rating with similarity value, achieves a better performance than some methods, especially CMN. However, the proposed method without rating conversion cannot overcome the SAMN method in any prediction accuracy metric, in Table 9.

Table 7 Comparison of the experimental results in terms of percentage between the proposed method without rating conversion and the other methods using nDCG

Datasets	Methods	nDCG@ <i>k</i>					
		5	10	20	30	40	50
ML-1M	SVD	+23.932%	-20.286%	-19.329%	-23.563%	-25.512%	-26.134%
	NMF	+30.484%	-11.429%	-14.398%	-18.582%	-20.857%	-21.053%
	CMN	+27.635%	+2.571%	+5.325%	+5.172%	+5.400%	+5.082%
	SAMN	+3.419%	-22.857%	-13.807%	-15.709%	-16.946%	-17.423%
Epinions	SVD	+0.417%	-4.881%	-7.341%	-2.319%	-	-
	NMF	+2.778%	-4.184%	-6.233%	-3.820%	-	-
	CMN	+3.056%	+2.232%	+6.094%	+4.366%	-	-
	SAMN	-3.472%	-7.671%	-5.263%	-0.682%	-	-
Yelp	SVD	+6.728%	+2.488%	-0.948%	-2.070%	-3.185%	-4.147%
	NMF	+9.480%	+4.666%	+0.632%	-0.637%	-0.955%	-1.435%
	CMN	+3.211%	+4.510%	+5.213%	+5.732%	+5.892%	+6.220%
	SAMN	+2.752%	-4.044%	-4.265%	-4.618%	+3.662%	+1.276%

Table 8 Comparison of the experimental results in terms of percentage between the proposed method without rating conversion and the other methods using HR

Datasets	Methods	HR@ <i>k</i>					
		5	10	20	30	40	50
ML-1M	SVD	+10.690%	-2.723%	-1.115%	-2.537%	-3.112%	-3.270%
	NMF	+17.586%	+4.208%	+1.433%	-0.149%	-1.132%	-2.044%
	CMN	+1.379%	+0.743%	+4.777%	+4.925%	+5.375%	+5.041%
	SAMN	-12.069%	-8.911%	+0.159%	-0.597%	-0.283%	-0.272%
Epinions	SVD	+0.388%	-0.729%	+1.523%	+0.919%	-	-
	NMF	+3.359%	+3.159%	+4.679%	+3.166%	-	-
	CMN	+1.809%	+1.580%	+4.135%	+1.736%	-	-

	<b>SAMN</b>	-3.618%	-2.430%	-0.435%	-0.102%	-	-
<b>Yelp</b>	<b>SVD</b>	+6.996%	+2.058%	+1.457%	+1.691%	+0.774%	+2.018%
	<b>NMF</b>	+8.779%	+3.429%	+2.517%	+2.601%	+1.935%	+3.026%
	<b>CMN</b>	+2.743%	0.000%	+0.132%	+0.650%	+0.258%	+1.639%
	<b>SAMN</b>	+19.753%	-0.412%	-3.046%	-3.901%	-3.484%	-4.666%

*Table 9 The experimental results comparison between the other methods and the proposed method without rating conversion in terms of percentage using prediction accuracy metrics*

<b>Datasets</b>	<b>Metrics</b>	<b>SVD</b>	<b>NMF</b>	<b>CMN</b>	<b>SAMN</b>
<b>ML-1M</b>	<b>Precision</b>	-2.177%	-0.768%	+1.921%	-0.896%
	<b>Recall</b>	-2.089%	-0.914%	+70.757%	-1.305%
	<b>RMSE</b>	+21.712%	+18.780%	-1.585%	+18.463%
<b>Epinions</b>	<b>Precision</b>	-5.632%	-3.159%	+1.236%	-6.044%
	<b>Recall</b>	+1.274%	+4.968%	+4.586%	-1.656%
	<b>RMSE</b>	+10.570%	+6.709%	-3.608%	+10.759%
<b>Yelp</b>	<b>Precision</b>	+7.920%	+9.456%	+0.709%	-0.355%
	<b>Recall</b>	-1.552%	+3.105%	+0.647%	-0.388%
	<b>RMSE</b>	+1.331%	-9.139%	-9.317%	+4.525%

When comparing the proposed method with and without rating conversion, the comparison results in percentage are shown in Table 10-12. Table 10 and 11 show the comparison of the experimental result of the proposed method with and without rating conversion in terms of percentage. Similar to the Table 7 and 8, the green numbers mean the proposed method obtain the higher performance than the proposed method without rating conversion. From Table 12, green numbers mean the proposed method can predict the rating more accurately than without rating conversion. Blue numbers in Table 12 mean the proposed method performs the prediction closer or less error than the proposed method without rating conversion. Therefore, the proposed method obtained better performance in terms of ranking prediction, accuracy and prediction error.

Table 10 Comparison of the experimental results in terms of percentage between the proposed method with and without rating conversion using nDCG

Datasets	nDCG@ <i>k</i>					
	5	10	20	30	40	50
ML-1M	+10.230%	+30.830%	0.000%	0.000%	0.000%	0.000%
Epinions	+3.872%	+8.429%	+6.839%	+6.386%	-	-
Yelp	+3.111%	+4.173%	+4.812%	+5.279%	+5.564%	+5.714%

Table 11 Comparison of the experimental results in terms of percentage between the proposed method with and without rating conversion using HR

Datasets	HR@ <i>k</i>					
	5	10	20	30	40	50
ML-1M	+13.947%	+11.209%	+3.976%	+3.597%	+3.547%	+3.548%
Epinions	+4.208%	+4.302%	+2.130%	+0.204%	-	-
Yelp	+7.955%	+7.487%	+3.822%	+4.234%	+3.846%	+5.257%

Table 12 The experimental results comparison between the proposed method with and without rating conversion in terms of percentage using prediction accuracy metrics

Datasets	Metrics	Without Rating Conversion	Proposed Method	%
ML-1M	Precision	0.781	0.811	+3.699%
	Recall	0.766	0.809	+5.315%
	RMSE	1.262	0.985	-28.122%
Epinions	Precision	0.728	0.789	+7.731%
	Recall	0.785	0.812	+3.325%
	RMSE	1.58	1.395	-13.262%
Yelp	Precision	0.846	0.866	+2.309%

	<b>Recall</b>	0.773	0.791	+2.276%
	<b>RMSE</b>	1.127	1.037	-8.679%

#### 4.1.4.2 ALL NEIGHBORS EXPERIMENTAL RESULTS

From selecting  $N$  neighbors as neighbors, the 30 neighbors achieve the highest performance compared with the outcomes from 10, 20, 40, and 50 neighbors. Therefore, the results of the 30 neighbors are employed to compare with the experimental results of using all neighbors in this subsection. Table 13 shows the comparing the nDCG results of the 30 neighbors ( $N = 30$ ) and all neighbors.

Table 13 The experimental results of using  $N$  neighbor and all neighbor

	@ $k$	ML-1M		Epinions		Yelp	
		N=30	All	N=30	All	N=30	All
<b>nDCG</b>	<b>5</b>	<b>0.391</b>	0.362	0.749	<b>0.756</b>	<b>0.675</b>	0.627
	<b>10</b>	<b>0.506</b>	0.435	0.783	<b>0.790</b>	<b>0.671</b>	0.663
	<b>20</b>	<b>0.675</b>	0.609	0.775	<b>0.809</b>	<b>0.665</b>	0.645
	<b>30</b>	<b>0.712</b>	0.640	0.783	<b>0.825</b>	<b>0.663</b>	0.598
	<b>40</b>	<b>0.742</b>	0.665	-	-	<b>0.665</b>	0.549
	<b>50</b>	<b>0.761</b>	0.685	-	-	<b>0.665</b>	0.543
<b>HR</b>	<b>5</b>	<b>0.337</b>	0.315	0.808	<b>0.811</b>	<b>0.792</b>	0.773
	<b>10</b>	<b>0.455</b>	0.438	0.860	<b>0.863</b>	<b>0.788</b>	0.766
	<b>20</b>	<b>0.654</b>	0.652	0.939	<b>0.940</b>	<b>0.785</b>	0.735
	<b>30</b>	<b>0.695</b>	0.686	0.981	<b>0.986</b>	<b>0.803</b>	0.782
	<b>40</b>	<b>0.733</b>	0.712	-	-	<b>0.806</b>	0.783
	<b>50</b>	<b>0.761</b>	0.734	-	-	<b>0.837</b>	0.822
<b>Precision</b>		<b>0.811</b>	0.787	0.789	<b>0.793</b>	<b>0.866</b>	0.852
<b>Recall</b>		<b>0.809</b>	0.803	0.812	<b>0.815</b>	<b>0.791</b>	0.784
<b>RMSE</b>		<b>0.985</b>	1.054	1.395	<b>1.341</b>	<b>1.037</b>	1.073

ML-1M and Yelp dataset results using 30 neighbors receive higher than all neighbors in all evaluation metrics. In contrast, the Epinions dataset with all neighbors obtained higher outcomes than using 30 neighbors. Figure 12 [a, c] show that almost similarity values of ML-1M and Yelp datasets are close to zero, which means almost all neighbors are less similar to the target users in both datasets. In contrast, in Figure 12 [b], almost all the similarity values in the Epinions dataset are close to one. Therefore, the neighbors from the Epinions dataset have a higher quality for predicting the recommendation than the ML-1M and Yelp datasets.

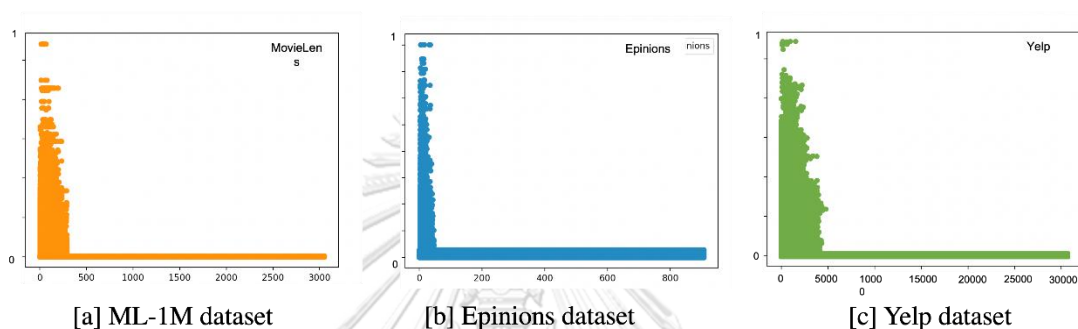


Figure 12 Similarity values between the neighbors and the target user of three datasets

## 4.2 SYNTATICDATASET

Due to the real-world datasets being unable to control the data distribution, the full rating matrices are generated to analyze the results using different rating distributions: the normal distribution, skewed right distribution, and skewed left distribution. The full rating matrix dataset is an ideal user-item dataset in the collaborative filtering technique. Each element in the matrix is the rating of user row  $i$  th on item column  $j$  th. Besides these three distributions, the partial rating matrix is also created. It is a dataset similar to the real-world dataset with normal rating distribution. To evaluate the performance of the synthetic datasets, the nDCG, HR, Precision, Recall, and RMSE metrics are utilized to assess the performance of the proposed method and the baseline researches.

### 4.2.1 DATA GENERATION SETTING

In this work, the rating range of full and partial rating matrices are fixed to range 1-5. In order to generate the datasets, there are four main parameters: the numbers of a user, item, rates, and rating score.

		Item				
User	1	2	3	4	5	
	4	3	1	5	2	
	3	1	5	2	4	
	2	5	4	1	3	
	5	4	2	3	1	

[a] Normal rating distribution

		Item				
User	1	4	3	5	5	
	5	3	5	4	2	
	3	4	3	5	5	
	5	4	3	2	1	
	5	5	3	1	4	

[b] Skewed right rating distribution

		Item				
User	1	3	1	2	1	
	1	1	2	4	5	
	3	2	1	1	1	
	2	1	2	4	1	
	1	5	1	2	2	

[c] Skewed left rating distribution

		Item				
User	2		4		1	
		1	3			
		2		3		
	4			1		
		4			5	

[d] Partial rating distribution

Figure 13 The example of synthetic datasets

There are three types of distributions of the full rating matrix that:

- **Normal rating distribution:** A dataset in which each rating score has the same number of ratings. Figure 13 [a] shows an example of a normal rating distribution with one rating for each rating score.
- **Skewed right rating distribution:** A dataset in which the number of high ratings outnumbers the number of low ratings. Figure 13 [b] shows a skewed right rating distribution in which the number of score 3, 4 and 5 is greater than the number of score 1 and 2.
- **Skewed left rating distribution:** A dataset that the low rating score exceeds the high rating score. Figure 13 [c] shows an example of skewed right rating distribution, which number of rating score 1 and 2 are more than score 3, 4 and 5.

Moreover, the size of each rating distribution depends on  $k$  of nDCG matrices to distinguish the effect of dataset size. In this work,  $nDCG@k$  for  $k \in \{5, 10, 20, 30, 40, 50\}$  are evaluated. Therefore, there are six datasets size in each rating distribution. The data of each dataset is divided into a ratio 80:20 for train:test sets. Suppose  $k = 5$ , the minimum number for user-item interaction that can be evaluated using  $k = 5$  is twenty five. Therefore, I use the number of users, items, raters, and a total number of rating scores equal to twenty five.

The normal distribution partial rating dataset is simulated, which is more similar to the real-world dataset than the full rating datasets. The partial dataset is the dataset that the users have rated some items. It contains more additional rating data than the full rating dataset. Figure 13 [d] shows an example of a partial rating dataset. The partial rating matrix is generated by using a 1000-by-1000 rating matrix and fifty raters per item.



The parameter settings for all synthetic datasets are shown in Table 14. The datasets gen1 to gen6 are the normal rating distribution with different sizes. The datasets gen7 to gen12 are the skewed right rating distribution, and datasets gen13 to gen18 are the skewed left. The dataset gen19 is the partial dataset that has two hundred and fifty raters similar to the gen6. However, gen19 can contain more rating patterns than the gen6 dataset because the target user does not need to rate all items in the dataset.

Table 14 The parameters of synthetic data

Dataset name	$k$	#user	#item	#rater	#1	#2	#3	#4	#5
gen1	5	25	25	25	5	5	5	5	5
gen2	10	50	50	50	10	10	10	10	10
gen3	20	100	100	100	20	20	20	20	20
gen4	30	150	150	150	30	30	30	30	30
gen5	40	200	200	200	40	40	40	40	40
gen6	50	250	250	250	50	50	50	50	50
gen7	5	25	25	25	2	3	5	7	8
gen8	10	50	50	50	3	5	10	15	17
gen9	20	100	100	100	5	10	20	30	35
gen10	30	150	150	150	10	15	30	45	50
gen11	40	200	200	200	10	20	40	60	70
gen12	50	250	250	250	15	20	50	80	85
gen13	5	25	25	25	8	7	5	3	2
gen14	10	50	50	50	17	15	10	5	3
gen15	20	100	100	100	35	30	20	10	5
gen16	30	150	150	150	50	45	30	15	10
gen17	40	200	200	200	70	60	40	20	10
gen18	50	250	250	250	85	80	50	20	15
gen19	50	1,000	1,000	250	50	50	50	50	50

#### 4.2.2 EXPERIMENTAL RESULTS

In this research, nineteen datasets are analyzed to investigate three distinct data distributions and dataset sizes. Table 15 presents the nDCG experimental results for all synthetic datasets. In terms of the ranking list, the proposed method obtained the highest nDCG results when compared to the baselines.

*Table 15 The nDCG experimental results of synthetic data*

Dataset name	$k$	SVD	NMF	CMN	SAMN	Proposed Method
gen1	5	0.716	0.757	0.758	0.748	<b>0.758</b>
gen2	10	0.695	0.681	0.695	0.695	<b>0.696</b>
gen3	20	0.623	0.640	0.640	0.638	<b>0.642</b>
gen4	30	0.613	0.611	0.615	0.614	<b>0.615</b>
gen5	40	0.581	0.589	0.595	0.594	<b>0.601</b>
gen6	50	0.578	0.78	0.579	0.578	<b>0.584</b>
gen7	5	0.779	0.794	0.816	0.814	<b>0.816</b>
gen8	10	0.742	0.750	0.751	0.743	<b>0.752</b>
gen9	20	0.705	0.695	0.712	0.711	<b>0.712</b>
gen10	30	0.694	0.684	0.695	0.694	<b>0.696</b>
gen11	40	0.677	0.679	0.680	0.680	<b>0.691</b>
gen12	50	0.682	0.682	0.687	0.688	<b>0.689</b>
gen13	5	0.745	0.742	0.746	0.746	<b>0.748</b>
gen14	10	0.663	0.688	0.693	0.695	<b>0.698</b>
gen15	20	0.632	0.630	0.643	0.647	<b>0.649</b>
gen16	30	0.589	0.595	0.583	0.588	<b>0.590</b>
gen17	40	0.578	0.580	0.586	0.589	<b>0.591</b>
gen18	50	0.558	0.553	0.566	0.567	<b>0.570</b>
gen19	50	0.674	0.570	0.564	0.573	<b>0.689</b>

In terms of data distribution, the skewed right rating distribution dataset obtained the highest nDCG results compared to the normal rating distribution and the skewed left rating distribution, according to the experimental results. Since most rating scores in the skewed right data distribution are high rating scores, these datasets have many positive preferences. In contrast, the skewed left rating distribution has a lot of negative preferences. Typically, making recommendations is more about suggesting things that people like rather than things that they dislike. Therefore, using positive preference is more powerful in recommendation than using negative preference. Each rating score in the normal rating distribution datasets has an equal number, implying that a number of positive preferences equal a number of negative preferences. As a result, normal rating distribution datasets produced lower nDCG results than skewed right rating distribution datasets and higher nDCG results than skewed left rating distribution datasets.

Because simulated datasets are dependent on  $k$ , the small dataset performs better in terms of nDCG than the large dataset. A small dataset mean using a small number of  $k$  to evaluate. A large set means using a large number of  $k$  to evaluate. Therefore, using a small number of  $k$  has a greater chance of ranking correctly than the large  $k$ .

Since the largest full normal distribution rating matrix is dataset gen6, which was used to evaluate in this study. To compare to gen6, gen19 is generated in a normal distribution with the same number of raters as gen6. From Table 15, the dataset gen19 obtained a higher ranking quality than gen6. Because gen19 contains more data in learning than gen6, the partial rating dataset has more varied patterns of interactions. Thus, gen19 outperforms gen6 in predicting the recommended item.

The HR metrics evaluate how many ratings in a  $k$  sized list of ranked things are matched with the actual rating ranked items by using rating prediction in the test set. Due to the number of interactions in the simulated datasets generated depending on  $k$ , the HR uses  $k$  predicted items divided by all predicted items, equal to  $k$ . Therefore, the HR result of all simulated datasets is equal to one.

When compared to two latent factor models and two current NCF with friends, the experimental results of the proposed method achieve the highest prediction results. The normal rating distribution dataset obtained lower error than the other rating distribution datasets in terms

of prediction accuracy in Table 16. Because the normal rating distribution dataset learns the equality of user's preference, the prediction range is more comprehensive than the other two distributions.

*Table 16 The accuracy experimental results of synthetic data*

<b>Dataset name</b>	<b><math>k</math></b>	<b>SVD</b>	<b>NMF</b>	<b>CMN</b>	<b>SAMN</b>	<b>Proposed Method</b>
<b>gen1</b>	<b>Precision</b>	0.508	0.585	0.391	0.391	<b>0.547</b>
	<b>Recall</b>	0.500	0.563	0.620	0.623	<b>0.625</b>
	<b>RMSE</b>	1.489	1.527	1.709	2.439	<b>1.220</b>
<b>gen2</b>	<b>Precision</b>	0.530	0.496	0.485	0.385	<b>0.539</b>
	<b>Recall</b>	0.502	0.453	0.388	0.619	<b>0.621</b>
	<b>RMSE</b>	1.438	1.686	1.729	2.403	<b>1.202</b>
<b>gen3</b>	<b>Precision</b>	0.492	0.512	0.357	0.357	<b>0.499</b>
	<b>Recall</b>	0.495	0.460	0.590	0.592	<b>0.597</b>
	<b>RMSE</b>	1.512	1.540	1.746	1.711	<b>0.873</b>
<b>gen4</b>	<b>Precision</b>	0.507	0.513	0.520	0.353	<b>0.525</b>
	<b>Recall</b>	0.480	0.467	0.516	0.594	<b>0.595</b>
	<b>RMSE</b>	1.509	1.487	1.434	1.638	<b>0.838</b>
<b>gen5</b>	<b>Precision</b>	0.506	0.510	0.518	0.354	<b>0.530</b>
	<b>Recall</b>	0.490	0.457	0.498	0.591	<b>0.594</b>
	<b>RMSE</b>	1.521	1.449	1.133	1.440	<b>1.031</b>
<b>gen6</b>	<b>Precision</b>	0.520	0.512	0.470	0.512	<b>0.517</b>
	<b>Recall</b>	0.495	0.482	0.392	0.521	<b>0.595</b>
	<b>RMSE</b>	1.540	1.446	1.350	1.380	<b>0.854</b>
<b>gen7</b>	<b>Precision</b>	0.640	0.623	0.586	0.586	<b>0.733</b>
	<b>Recall</b>	0.500	0.547	0.762	0.763	<b>0.766</b>
	<b>RMSE</b>	1.428	1.603	1.387	1.374	<b>1.372</b>
<b>gen8</b>	<b>Precision</b>	0.727	0.737	0.696	0.716	<b>0.857</b>
	<b>Recall</b>	0.549	0.571	0.822	0.825	<b>0.828</b>

	<b>RMSE</b>	1.230	1.288	1.237	1.726	<b>1.208</b>
<b>gen9</b>	<b>Precision</b>	0.732	0.727	0.709	0.709	<b>0.887</b>
	<b>Recall</b>	0.570	0.603	0.837	0.840	<b>0.842</b>
	<b>RMSE</b>	1.207	1.241	1.582	1.522	<b>1.184</b>
<b>gen10</b>	<b>Precision</b>	0.723	0.725	0.730	0.700	<b>0.897</b>
	<b>Recall</b>	0.500	0.404	0.511	0.513	<b>0.528</b>
	<b>RMSE</b>	1.251	1.238	1.695	1.734	<b>1.173</b>
<b>gen11</b>	<b>Precision</b>	0.745	0.741	0.723	0.753	<b>0.904</b>
	<b>Recall</b>	0.592	0.524	0.530	0.500	<b>0.549</b>
	<b>RMSE</b>	1.227	1.201	1.260	1.220	<b>1.143</b>
<b>gen12</b>	<b>Precision</b>	1.760	0.763	0.757	0.754	<b>0.868</b>
	<b>Recall</b>	0.608	0.640	0.705	0.841	<b>0.850</b>
	<b>RMSE</b>	1.227	1.176	1.212	1.359	<b>1.092</b>
<b>gen13</b>	<b>Precision</b>	0.493	0.440	0.453	0.493	<b>0.502</b>
	<b>Recall</b>	0.500	0.484	0.591	0.571	<b>0.608</b>
	<b>RMSE</b>	1.241	1.528	1.217	1.218	<b>1.214</b>
<b>gen14</b>	<b>Precision</b>	0.495	0.538	0.534	0.534	<b>0.610</b>
	<b>Recall</b>	0.504	0.580	0.566	0.566	<b>0.586</b>
	<b>RMSE</b>	1.247	1.358	1.238	1.238	<b>1.217</b>
<b>gen15</b>	<b>Precision</b>	0.538	0.541	0.544	0.564	<b>0.616</b>
	<b>Recall</b>	0.559	0.580	0.552	0.572	<b>0.593</b>
	<b>RMSE</b>	1.193	1.250	1.276	1.271	<b>1.172</b>
<b>gen16</b>	<b>Precision</b>	0.544	0.531	0.534	0.564	<b>0.556</b>
	<b>Recall</b>	0.502	0.565	0.563	0.566	<b>0.576</b>
	<b>RMSE</b>	1.262	1.268	1.258	1.252	<b>1.179</b>
<b>gen17</b>	<b>Precision</b>	0.534	0.543	0.553	0.524	<b>0.636</b>
	<b>Recall</b>	0.534	0.571	0.645	0.652	<b>0.708</b>
	<b>RMSE</b>	1.233	1.201	1.677	1.594	<b>1.212</b>
<b>gen18</b>	<b>Precision</b>	0.552	0.539	0.514	0.534	<b>0.660</b>

	<b>Recall</b>	0.530	0.579	0.538	0.549	<b>1.619</b>
	<b>RMSE</b>	1.224	1.181	1.264	1.264	<b>1.128</b>
<b>gen19</b>	<b>Precision</b>	0.517	0.505	0.357	0.537	<b>0.535</b>
	<b>Recall</b>	0.531	1.525	0.597	0.597	<b>0.896</b>
	<b>RMSE</b>	1.212	1.269	1.416	1.434	<b>1.201</b>



## CHAPTER 5 DISCUSSIONS

Because the concept of neighbors is applied to a neural network, two issues need to be considered: similarities between the neighbors and the target user and rating conversion. Therefore, this work proposed the similarity and rating conversion modules, which can deal with both issues. Furthermore, the proposed method classifies neighbor selection into two types: N neighbors and all neighbors. Additionally, the proposed method is compared to the present NCF with friends and the latent factor model in the experiment.

### 5.1 PERFORMANCE OF SIMILARITY MODULE

The similarity between the neighbor and the target user indicates how much neighbors and the target user are similar. This similarity value is one component of the prediction equation of the collaborative filtering technique (Equation 2), which is combined with the neighbor's ratings using a weighted average on the neighbors' ratings where these similarity values are used as weights. The neighbors' rating scores are the preference of neighbors who are similar to the target user. To create the similarity between the neighbor and the target user, CMN performs the similarity value by combining neighbor embedding, target user embedding, and target item embedding together. SAMN creates the similarity value using joint embedding between neighbor embedding and target user embedding. In comparison, the proposed method performs the similarity value by applying attention between users, which are dot products between neighbor embedding and the target user embedding. In terms of usage, the current NCFs with friends compute the similarity value and integrate these similarity values with the user's representation rating to perform the target user's predicted rating. Therefore, both current NCFs with friends do not use the neighbors' similarity to combine with the neighbors' ratings as the CF technique's rating prediction equation. In comparison, the proposed method uses the attention of neighbors to compute the similarity levels in the fourth step and the target user embedding and to normalize into the range  $[0,1]$ . After that, these similarities between the neighbors and the target user are used as weights on the neighbor's rating for computing weighted average in the rating prediction step. Therefore, the proposed method provides a more accurate prediction than the current NCFs with friends.

## 5.2 PERFORMANCE OF RATING CONVERSION MODULE

According to the collaborative filtering concept, the neighbors' preferences are used to predict the target user's ratings. Actually in the real-world, a different user has a different rating range, although they have similar preferences. For example, user U1 gives a rating in the range from 3 to 5 for "dislike" to "like," while user U3 gives a rating in range from 1 to 3 instead. It means "like" of user U3 equals "dislike" of user U1. To predict the rating of item i3 to a user to user U1 using user U3, user U3 will suggest rating 3 to user U1, which is "like" for user U3. However, rating 3 for user U1 means "dislike." Therefore, the rating range of user rating needs to be considered. However, It is one of the issues in the collaborative filtering approach that is not often discussed.

		Items					Rating range
		i1	i2	i3	i4	i5	
Users	U1	5	4	?	3	4	3 - 5
	U2	4	5	3	3	5	3 - 5
	U3	3	2	3	1	2	1 - 3
	U4	3	2	3	4	2	2 - 4

Figure 14 The example of Rating range issue

There was an idea to use rating normalization to solve this issue in the past. However, using normalization to convert a user's rating is converting based on the original user's rating only. For example (Figure 15), the target users A1 and A2 have different rating ranges but the same neighbor. The target user A1 usually rates the item at score 2. In contrast, the target user A2 usually rates at score 4. If both target users have the same neighbor and the predicted rating score from the neighbor on an item is 4. However, that means the target user A1 much more strongly prefers this item than the target user A2. Because the rating score 4 is "normally" for the target user A2, while the rating score 4 means "really like" for the target user A1. Therefore, solving the rating range issue using rating normalization is ineffective.



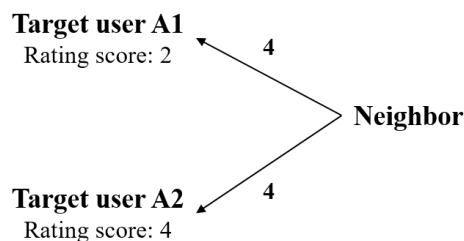


Figure 15 The example of rating normalization problem

For this reason, converting the rating range of neighbor into the target user's rating range was proposed. Thus, the proposed method applied this idea with the region embedding to solve the rating conversion issue. From the experimental results, the effectiveness of the rating conversion module is evaluated by removing converting neighbors into the target user aspect from this module. The experimental results from Tables 4-6 shown that the proposed method obtained the highest efficiency since the neighbors' preferences are converted into the target user aspect.

Current NCFs with friends perform the predicted ratings employing users' representations, items' representations, and the neighbor's influences without concern the rating conversion issue. While the proposed method concerns and solves the rating conversion issue in the rating conversion module. This module converts the neighbors' rating ranges into the target user aspect by element-wise multiplication between the target user's LCU and the neighbors' representations. Afterward, the neighbors' rating in the target user's perspective view is created using the converted neighbors' representation.

จุฬาลงกรณ์มหาวิทยาลัย  
CHULALONGKORN UNIVERSITY

### 5.3 NEIGHBOR SELECTION

In the traditional collaborative filtering technique, the neighbor's rating and the similarity between the neighbor and the target user are integrated to perform the rating prediction. Therefore, the number of neighbors is essential in collaborative filtering research. Using all raters as neighbors may lead to an incorrect prediction because of the low-quality neighbors. In order to evaluate the number of neighbors, employing neighbors is divided into two types:  $N$  neighbors and all neighbors. The experimental results in Table 7 show that using all neighbors to perform the predicted rating obtained a lower performance than using  $N$  neighbors on ML-1M and Yelp datasets. In both datasets, the similarity value of most neighbors is close to zero, which means there

are many low-quality neighbors. Therefore, using all neighbors on ML-1M and Yelp datasets performs lower accuracy than  $N$  neighbors in prediction.

The results of the Epinions dataset utilizing all neighbors are higher than using  $N$  neighbors. Almost all of the similarity values in Figure 12 are close to one. Therefore, most of the neighbors in the Epinions dataset are of high quality. Hence, using all neighbors can improve the prediction results. To summarize, two of three in real-world datasets use  $N$  neighbors better performance than all neighbors because there are many low-quality neighbors in the datasets. If the dataset has a large number of high-quality neighbors, employing all neighbors outperforms using  $N$  neighbors.

#### 5.4 DATA SIMULATION

The full rating dataset and partial rating dataset are the two types of synthetic datasets used in this study. When analyzing the full rating dataset and the partial rating dataset in a normal distribution, the partial rating dataset performs better than the full rating dataset. The reason is that the partial rating dataset contains more rating patterns than the full rating dataset. For example, there are three items in the dataset. In the case of a full rating dataset with normal distribution, six rating patterns can occur in each user. While the partial rating dataset with a normal distribution, twenty-four rating patterns can occur because the users do not need to rate all items in the dataset. Hence, the model can learn various rating patterns in the partial rating dataset, which leads the partial rating dataset to outperform the full rating dataset.

The rating distribution and dataset sizes are considered in the full rating dataset. This work simulates three types of rating distribution: the normal rating distribution, skewed right rating distribution, and skewed left rating distribution. The skewed right rating distribution outperforms the other two distributions, as seen in Table 9. Due to the recommendation nature, the positive preferences of users are utilized to perform the suggestion more than negative preferences.

Under the distribution condition, the various size of the rating matrix depending on  $k$  of nDCG are simulated. The results showed that the small dataset performed better than the large dataset. Because the small dataset uses the less  $k$  to rank and the large dataset uses the large  $k$  to rank. The less  $k$  has more chance to rank the predicted rating correctly than the large  $k$ . Figure 16

shows the example of comparing  $k=5$  and  $k=10$ . It can be seen that using  $k=5$  has a higher probability of raking correctly than  $k=10$ .

$k = 5$	Actual item list	i1	<b>i2</b>	<b>i3</b>	i4	i5
	Predicted item list	i1	<b>i3</b>	<b>i2</b>	i4	i5

$k = 10$	Actual item list	i1	<b>i2</b>	<b>i3</b>	i4	<b>i5</b>	<b>i6</b>	<b>i7</b>	i8	i9	i10
	Predicted item list	i1	<b>i3</b>	<b>i2</b>	i4	<b>i6</b>	<b>i7</b>	<b>i5</b>	i8	i9	i10

*Figure 16 The example of comparing k ranking*

In order to compare the proposed method with other methods, the proposed method outperforms all the current NCF with friends, CMN and SAMN, and the latent factor models, SVD and NMF. Compared with the current NCF with friends, the CMN and SAMN create only similarities between the neighbor and the target user. In comparison, the proposed method computes the similarity between the neighbor and the target user and applies the converted neighbors' ratings in the prediction process, as discussed in Sections 5.1 and 5.2. In the case of the latent factor models, the reasons are the same as using real-world datasets, which are discussed in Section 5.5.

## 5.5 COMPARING METHOD WITH LATENT FACTOR MODEL

To evaluate the proposed method with the latent factor model, the SVD and NMF are used as the experiment baselines, assessing a real-world and simulated dataset. The experimental outcomes show that the proposed method performs better than both SVD and NMF. According to previous works, NCF [16] and current NCFs with friends perform better than the matrix factorization, one of the latent factor models. In this work, I want to prove whether the proposed method is better than the latent factor model. Because using a neural network make the model can capture the non-linear data distribution and more latent dimension than the latent factor model.

## CHAPTER 6 CONCLUSIONS

The considering the similarity and the rating conversion of neighbors on NCF strategies are proposed. It applies neighbors' concept of collaborative filtering to a neural network. Besides, the proposed method can also handle the similarity between the neighbors and the target user and rating conversion issues. Different users have different preference ranges, which leads to inaccurate predictions. Therefore, the rating conversion should be considered. One crucial component used to obtain the rating prediction in the collaborative filtering technique is the similarity between the neighbors and the target user.

The target users' rating prediction is the result of the proposed method. It is obtained using a weighted average on the neighbors' ratings where the similarities levels are used as weights according to the rating prediction of the CF technique. These similarity values are computed by utilizing the neighbor's attention via a dot product between the target user and the neighbor representation. The neighbors' ratings are performed applying a matrix factorization concept between the item's characteristics and the converted neighbors' representations.

In the experiment, the proposed method is evaluated on the real-world datasets and the simulation datasets and compared with existing NCFs with friends methods and the latent factor models. The results show that the proposed method outperforms all baselines using ranking-based evaluation and prediction accuracy metrics. The proposed method without rating conversion is compared to the proposed method to evaluate the rating conversion efficiency in the real-world dataset. The results demonstrate that the proposed method achieves higher efficiency than the proposed method without the rating conversion. In order to know the effectiveness of the number of neighbors, this work divided employing neighbors into two types: the  $N$  neighbors and all neighbors. The experimental result confirms that using  $N$  neighbors obtained better performance than using all neighbors in a large real-world dataset. Moreover, the effectiveness of using different rating distribution and datasets sizes are assessed. In terms of rating distribution, the skewed right rating distribution achieves the best performance in synthetic datasets. In terms of dataset size, the small datasets perform better than the large datasets.

From Figure 7, the number of neighbors must be equal for every target user because of the limitation of neural network implementation. Therefore, the maximum number of neighbors to learn the model need to be fixed. However, in reality, different target users do not necessitate having

the same number of neighbors. In the future, if a neural network model with the actual number of neighbors can be implemented, it may enhance performance and make the model more realistic.





จุฬาลงกรณ์มหาวิทยาลัย  
**CHULALONGKORN UNIVERSITY**

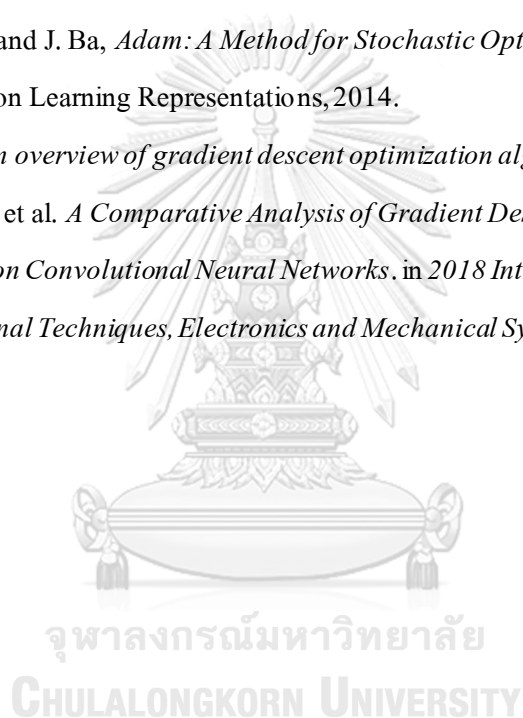
## REFERENCES

1. Isinkaye, F.O., Y.O. Folajimi, and B.A. Ojokoh, *Recommendation systems: Principles, methods and evaluation*. Egyptian Informatics Journal, 2015. **16**(3): p. 261-273.
2. Burke, R., A. Felfernig, and M.H. Göker, *Recommender Systems: An Overview*, in *Ai Magazine*. 2011. p. 6.
3. Nilashi, M., et al., *Collaborative Filtering Recommender Systems*. Research Journal of Applied Sciences, Engineering and Technology, 2013: p. 4168-4182.
4. Li, Q. and B.M. Kim, *An approach for combining content-based and collaborative filters*, in *Proceedings of the sixth international workshop on Information retrieval with Asian languages - Volume 11*. 2003, Association for Computational Linguistics: Sapporo, Japan. p. 17-24.
5. Bokde, D., S. Girase, and D. Mukhopadhyay, *Matrix Factorization Model in Collaborative Filtering Algorithms: A Survey*. Procedia Computer Science, 2015. **49**: p. 136-146.
6. Koren, Y., R. Bell, and C. Volinsky, *Matrix Factorization Techniques for Recommender Systems*. Computer, 2009. **42**(8): p. 30-37.
7. Shah, S., *Introduction to Matrix Factorization for Recommender Systems*. 2018.
8. Baker, K., *Singular Value Decomposition Tutorial*. 2013. **2005**.
9. Richardson, M., *Principal Component Analysis*. 2009.
10. Blei, D.M., A.Y. Ng, and M.I. Jordan, *Latent dirichlet allocation*. J. Mach. Learn. Res., 2003. **3**(null): p. 993-1022.
11. Albawi, S., T.A. Mohammed, and S. Al-Zawi. *Understanding of a convolutional neural network*. in *2017 International Conference on Engineering and Technology (ICET)*. 2017.
12. Liu, D. and G. Singh. *A Recurrent Neural Network Based Recommendation System*. 2016.
13. Gers, F.A. and E. Schmidhuber, *LSTM recurrent networks learn simple context-free and context-sensitive languages*. IEEE Transactions on Neural Networks, 2001. **12**(6): p. 1333-1340.
14. Gers, F.A., J. Schmidhuber, and F. Cummins. *Learning to forget: continual prediction with LSTM*. in *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*. 1999.

15. Hochreiter, S. and J. Schmidhuber, *Long Short-term Memory*. *Neural computation*, 1997. **9**: p. 1735-80.
16. He, X., et al., *Neural Collaborative Filtering*, in *Proceedings of the 26th International Conference on World Wide Web*. 2017, International World Wide Web Conferences Steering Committee: Perth, Australia. p. 173–182.
17. He, X., et al., *Outer product-based neural collaborative filtering*, in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. 2018, AAAI Press: Stockholm, Sweden. p. 2227–2233.
18. Mao, Y., et al., *TCR: Temporal-CNN for Reviews Based Recommendation System*, in *Proceedings of the 2018 2nd International Conference on Deep Learning Technologies*. 2018, Association for Computing Machinery: Chongqing, China. p. 71–75.
19. Bai, T., et al., *A Neural Collaborative Filtering Model with Interaction-based Neighborhood*, in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 2017, Association for Computing Machinery: Singapore, Singapore. p. 1979–1982.
20. Xi, W.-D., et al., *BPAM: recommendation based on BP neural network with attention mechanism*, in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. 2019, AAAI Press: Macao, China. p. 3905–3911.
21. Ebesu, T., B. Shen, and Y. Fang, *Collaborative Memory Network for Recommendation Systems*, in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 2018, Association for Computing Machinery: Ann Arbor, MI, USA. p. 515–524.
22. Chen, C., et al., *Social Attentional Memory Network: Modeling Aspect- and Friend-Level Differences in Recommendation*, in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 2019, Association for Computing Machinery: Melbourne VIC, Australia. p. 177–185.
23. Joulin, A., et al., *Bag of Tricks for Efficient Text Classification*. 2017. 427-431.
24. Mikolov, T., et al., *Distributed representations of words and phrases and their compositionality*, in *Proceedings of the 26th International Conference on Neural*



- Information Processing Systems - Volume 2*. 2013, Curran Associates Inc.: Lake Tahoe, Nevada, p. 3111–3119.
25. Chalermponpong, W., S. Maneeroj, and T. Atsuhiro. *Rating Pattern Formation for Better Recommendation*. in *2013 24th International Workshop on Database and Expert Systems Applications*. 2013.
  26. Qiao, C., et al., *A NEW METHOD OF REGION EMBEDDING FOR TEXT CLASSIFICATION*. 6th International Conference on Learning Representations (ICLR2018), 2018: p. 1-12.
  27. Kingma, D. and J. Ba, *Adam: A Method for Stochastic Optimization*. International Conference on Learning Representations, 2014.
  28. Ruder, S., *An overview of gradient descent optimization algorithms*. 2016.
  29. Dogo, E.M., et al. *A Comparative Analysis of Gradient Descent-Based Optimization Algorithms on Convolutional Neural Networks*. in *2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*. 2018.





จุฬาลงกรณ์มหาวิทยาลัย  
**CHULALONGKORN UNIVERSITY**

## VITA

**NAME** Thitiporn neammanee

**DATE OF BIRTH** 8 March 1995

**PLACE OF BIRTH** Bangkok, Thailand.

**INSTITUTIONS ATTENDED** Faculty of Science, Chulalongkorn University

**HOME ADDRESS** 24/148 Kasemsan1 Condo, Rama1 Road, Wangmai district,  
Patumwan, Bangkok, Thailand 10330

**PUBLICATION** Time-Aware Recommendation Based on User Preference Driven

