Customer-based Recommendation Model for Next Merchant Recommendation

Bayartsetseg Kalina, Ju-Hong Lee

Abstract

In the recommendation system of the credit card company, it is necessary to understand the customer patterns to predict a customer's next merchant based on their histories. The data we want to model is much more complex and there are various patterns that customers choose. In such a situation, it is necessary to use an effective model that not only shows the relevance of the merchants, but also the relevance of the customers relative to these merchants. The proposed model aims to predict the next merchant for the customer. To improve prediction performance, we propose a novel model, called Customer-based Recommendation Model (CRM), to produce a more efficient representation of customers. For the next merchant recommendation system, we use a synthetic credit card usage dataset, BC'17. To demonstrate the applicability of the proposed model, we also apply it to the next item recommendation with another real-world transaction dataset, IJCAI'16.

Keywords : Next Merchant Recommendation | Customer-based Recommendation | Deep Learning

I. INTRODUCTION

Credit card companies are looking for more people to use their services. A recommendation system is a simple way that their customers actively use their cards. As the variety and number of merchants selected by customers are growing dramatically, an important challenge is to provide customers with personalized recommendations for selecting merchants based on their preferences and needs. A similar challenge exists in the next item recommendation. For the next item recommendation, deep learning models have demonstrated the potential to learn more complex user-item interactions from training data[1,4-6,8,10-11,14]. GRU4Rec [6] used the GRU-based RNN for session-based recommendations. The recommender systems get to know the user, then recommend the specific next item under a transactional context by analyzing intra-transaction dependencies[14]. Several works are using max pooling or average pooling techniques to generate the latent vector representations of the subset of items[7,8]. While using max pooling and average pooling techniques, some information about the interactions of items may be lost. To overcome this disadvantage, the Attention-based Transactional context Embedding Model (ATEM) [14] uses an attention mechanism instead of pooling operations because of the relevance and transition between items in a

* This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2021R1F1A1050120)

Manuscript: 2023. 02. 06
Revised: 2023. 03. 03, 2023. 04. 07
Confirmation of Publication: 2023.5.12
Corresponding Author: Bayartsetseg Kalina
e-mail: kb0422.bk@gmail.com

transactional context. However, ATEM only uses item characteristics to learn transactional context vectors. In this paper, we first propose a novel model for the next merchant recommendation in credit card companies. The proposed model can produce a more efficient representation of customers for the next merchant recommendation. Customer similarity is important for the representation of the customer. We use a Euclidean distance of embedding vectors to measure customer similarity. Consequently, the learned transactional context vector is more efficient because it uses both customer and merchant characteristics. We use next-item recommendation models as benchmarks. The main contributions of this work are as follows.

- We introduce the next merchant recommendation model for credit card companies.
- The proposed model can produce a more efficient representation of customers with a joint probability and a regularization term of the loss function.
- Transactional context vectors, based on both customer and merchant characteristics, are more efficient than ATEM ones.
- We empirically show that our proposed model outperforms other benchmark models.

II. RELATED WORK

The YouTube recommender system [8] is comprised of two neural networks, candidate generation, and ranking. The candidate generation network retrieves a small subset of videos from the enormous YouTube corpus as a multi-class classification problem, with the user's activity history as input. The ranking network retrieves a few hundred videos from the outputs of the candidate generation network using external features describing the video and user. This system is one of the most successful applications in recommender systems. However, using a simple average pooling technique for a context is insufficient to predict the next item.

ATEM [14] showed the importance of the next item recommendation that could be misdirected by irrelevant items in a transactional context. For example[14], let {milk, apple, orange, bread} be a transaction. We want to predict the next item {bread} given the transactional context {milk, apple, orange}. If we assume that all items in the transactional context play the same role, then the predicted next item is not bread, but might be a vegetable due to the nearest contextual items. To address this problem, ATEM uses an attention layer to weigh the items in a transactional context instead of using average pooling or max pooling techniques. ATEM is not a personalized model. There are several personalized recommendation models such as [8,11,12] using both user and item embedding vectors. Customer characteristics should also be considered to learn more useful transactional context vectors. To do

this, we propose a customer-based recommendation model (CRM).

Lately, recommender systems have adopted recurrent neural networks (RNNs) [6,7,10] and transformer-based models [14,15] to capture the temporal dependencies for users and items. Long short-term memory (LSTM) is a more general and successful case for certain recurrent architectures that can be learned from classifying and predicting time series with time delays. One of the popular recommendation models, GRU4Rec[6] used the GRU-based RNN[3] for session-based recommendations.

III. PROPOSED METHOD

Our goal is to develop a model that predicts the next merchant of the customer based on their historical activities. Assume that we have sets of merchants and customers, denoted by M and C, respectively. The whole merchant set, $M = \{m_1, m_2, \dots, m_n\}$, has n merchants, and the whole customer set, $C = \{c_1, c_2, \dots, c_r\}$, has r customers. Let $T = \{t_1, t_2, \dots, t_{|T|}\}$ be the transactional context set. $|T|$ denotes the number of transactional contexts. Each transactional context consists of a subset of the merchants, $t = \{m_1, m_2, \dots, m_{k-1}\}$ containing $k-1$ merchants. We aim to predict the next merchant m_k.

The proposed model, CRM, is trained to learn the joint probability of merchant m_k and customer c_j given a transactional context t.

$$Pr(m_k, c_j|t) = Pr(m_k|c_j, t) \cdot Pr(c_j|t) \quad (1)$$

We use two conditional probabilities, $Pr(m_k|c_j, t)$ and $Pr(c_j|t)$. The network learned by conditional probability $Pr(m_k|c_j, t)$ is called *Merchant* and the second one is called *Customer*. Using the *Customer* network, we can produce a more efficient representation of the customer than only using an embedding layer. In other words, with the help of the *Customer* network, we can obtain customer representations that contain more detailed information about their purchases.

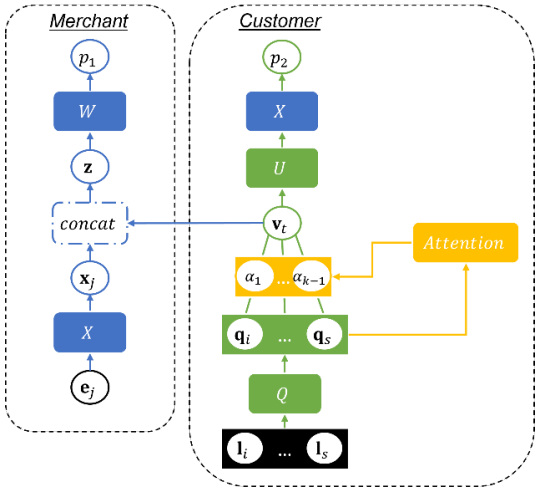

Fig. 1. The general framework of CRM

The general framework of CRM is illustrated in Figure 1. We will empirically show that the proposed model is more effective for the next merchant recommendation. The Softmax layer is used for computing the probabilities of the next merchant, $Pr(m_k|c_j, t)$ and the customer, $Pr(c_j|t)$. The attention layer [13] is used to obtain the embedding vector of the transactional context, t.

The *Customer* network is constructed to estimate a conditional probability

distribution $Pr(c_j|t)$. We use an embedding layer to map the one-hot encoding of the merchant into a real-valued embedding vector [10,11,13,14].

$$\mathbf{q}_s = Q \cdot \mathbf{l}_s \quad (2)$$

where $\mathbf{l}_s \in \mathbb{R}^n$ is the one-hot encoding of merchant m_s; $\mathbf{q}_s \in \mathbb{R}^d$ and $Q \in \mathbb{R}^{d \times n}$ are the embedding vector of the merchant m_s and the latent factor matrix that maps the one-hot encodings of the merchant into the real-valued embedding vectors, respectively.

We obtain the embedding vector of the transactional context using the Attention layer [10].

$$\mathbf{v}_t = \sum_{m_s \in t} \alpha_{sk} \mathbf{q}_s \quad \text{s.t.} \sum_{m_s \in t} \alpha_{sk} = 1 \quad (3)$$

where $\mathbf{v}_t \in \mathbb{R}^d$ and α_{sk} are the embedding vector of transactional context t and the integration weight of the merchant m_s in the context about target merchant m_k, respectively. The output layer formulation is given as Equation 4.

$$\hat{\mathbf{y}}_c = f\big(X \cdot f(U \cdot \mathbf{v}_t + \mathbf{b}_c)\big) \quad (4)$$

where $X \in \mathbb{R}^{r \times h}$ and $U \in \mathbb{R}^{h \times d}$ are the customer embedding and hidden layer weight matrices, respectively; and $\mathbf{b}_c \in \mathbb{R}^h$ denotes the bias vector; and f and $\hat{\mathbf{y}}_c \in \mathbb{R}^r$ are an activation function and a score vector of the customers, respectively. It is called *Customer* network because of estimating the conditional probability of a customer.

The *Merchant* network is constructed as a probabilistic classifier, which is learned to predict a probability distribution $Pr(m_k|c_j, t)$. This network input is the concatenated vector of the customer and transactional context.

We use an embedding layer to map the one-hot encoding of the customer, $\mathbf{e}_j$, into a real-valued embedding vector, $\mathbf{x}_j$.

The transactional context and customer embedding vectors, $\mathbf{v}_t$ and $\mathbf{x}_j$, are shared with the *Customer* network. The formulation of the output layer *Merchant* network is given as follows:

$$\hat{\mathbf{y}}_m = f\left(W \cdot \begin{bmatrix}\mathbf{v}_t \\ \mathbf{x}_j\end{bmatrix} + \mathbf{b}_m\right) \quad (5)$$

where $\hat{\mathbf{y}}_m \in \mathbb{R}^n$ is a score vector of the merchants; $W \in \mathbb{R}^{n \times (h+d)}$ and $\mathbf{b}_m \in \mathbb{R}^n$ are output layer weight matrix and bias vector, respectively.

For customer c_j and transactional context t, we define the probability of the next merchant m_k by the following Softmax function.

$$\hat{p}_1(m_k|c_j, t) = \frac{\exp(\mathbf{w}_k \cdot \mathbf{z})}{\sum_{j=1}^{n} \exp(\mathbf{w}_j \cdot \mathbf{z})} \quad (6)$$

where $\mathbf{w}_k \in \mathbb{R}^{h+d}$ is the k^{th} row vector of the output weight matrix W and $\mathbf{z} \in \mathbb{R}^{h+d}$ is a concatenated vector of transactional context and customer embedding. We also define the probability of customer c_j by the Softmax function. CRM chooses a merchant that maximizes the joint probability, $P(m_k, c_j|t)$. It also minimizes the distance of similar customers using a regularization term. So that we use the cross-entropy loss function with regularization term which is defined as:

$$L = -\sum_{m_k \in M} q_{kj} \cdot \log \hat{p}_{kj} + \lambda \left\| \mathbf{x}_j - \mathbf{x}_s \right\|_2^2 \quad (7)$$

where $\hat{p}_{kj}$ is the probability of the proposed model and q_{kj} is a target probability distribution that the customer c_j will choose merchant m_k, $\mathbf{x}_j$ is the embedding vector of the target customer, $\mathbf{x}_s$ is the embedding vector of the customer with the highest probability of the *Customer* network. λ is a hyperparameter that weights the importance of customer similarity.

IV. EXPERIMENTS

We evaluate our model on a synthetic credit card usage dataset, BC'17, for the next merchant recommendation and on a real-world transaction dataset, IJCAI'16, for the next item recommendation.

Dataset

BC'17. This is a synthetic transaction dataset generated from the distribution of the customer, merchant, sale date, and sale amount between January 2017 and December 2017 provided by a credit card company. In this dataset, a transaction means that a customer bought products or services from a merchant. For this dataset, we aim to predict the next merchant, m_6, for a customer c_j and his or her each set of five transactions, $t = \{m_1, m_2, m_3, m_4, m_5\}$

IJCAI'16. To demonstrate the applicability of PANM, we also use it for the next item recommendation. We use a real-world transaction dataset, IJCAI'16 for the next item recommendation. This dataset contains shopping transactions between July 2015 and November 2015 accumulated on Tmall/Taobao.com and the app Alipay. Like ATEM[14], we aim to predict the next item i_k using a transactional context $\tilde{t} = t \backslash i_k$ and user u_j. We present the statistics of the experimental datasets in Table 1.

Table 1. The statistics of the experimental datasets

	BC'17	IJCAI'16
#Customers/users	73,584	15,907
#Merchants/items	84,161	21,866
#Training data	1,296,664	524,725
#Test data	367,920	58,302

Benchmark models

DNN. The traditional DNN model[8,12] recommends the next merchant given a transactional context considered as a classification problem. We assess the performance of the DNN model by considering the average pooling for merchant embedding vectors in the transactions.

GRU4Rec. GRU4Rec[6] recommender is a deep RNN model which consists of gated recurrent units as session-based RNNs.

ATEM. ATEM[14] recommends the next merchant given a transactional context t using an attention layer. This model predicts the next merchant for a given transactional context, not for a customer.

Personalized ATEM. This model is computed with the conditional probability of the merchant given a customer c_j and a transactional context t, $Pr(m_k|c_j, t)$. We use an embedding layer to embed customers.

Evaluation metrics

The comparison models will rank the merchant list and then return the $top@k$ merchants as recommendations. To evaluate the performance of our model, we

used $Recall@k$ and $Precision@k$ to calculate successfully recommended merchants. $Precision@k$ is the proportion of recommended items in the $top@k$ set that are relevant.

$$Precision@k = \frac{\sum_{j=1}^{r} \left| Y_{c_j} \cap \hat{Y}_{c_j} \right|}{\sum_{j=1}^{r} \left| \hat{Y}_{c_j} \right|} \quad (8)$$

where r is the number of customers; Y_{c_j} and $\hat{Y}_{c_j}$ are the recommended items and the target items for customer c_j, respectively.

$Recall@k$ is the proportion of relevant items found in the $top@k$ recommendations.

$$Recall@k = \frac{\sum_{j=1}^{r} \left| Y_{c_j} \cap \hat{Y}_{c_j} \right|}{\sum_{j=1}^{r} \left| Y_{c_j} \right|} \quad (9)$$

To evaluate ranking quality of recommended items, we used $MRR@k$ (Mean Reciprocal Rank) which is a rank-aware evaluation metric, widely used in recommendation systems.

$$MRR@k = \frac{1}{r} \sum_{j=1}^{r} \frac{1}{rank_{c_j}} \quad (10)$$

where $rank_{c_j}$ refers to the rank position of the relevant items for customer c_j. There is a trade-off between precision and recall. We chose $k = 10$ for the BC'17 dataset and $k = 3$ for the IJCAI'16 dataset.

Experiments

We ran experiments using two datasets, BC'17 and IJCAI'16. For the BC'17 dataset, we aim to predict the next merchant of five sequential transactions. The embedding dimensions are set to $\{150, 200, 250, 300\}$ for customers and merchants. For this dataset, we hold out the last five transactions as the test set for each customer and other data as the training set. We adopt the Adam optimizer[6] with the learning rate set to 0.001 and the batch size of 100 for optimizing the loss function. For this dataset, a hyperparameter in the loss function, λ, is set to 0.9. It means that customer similarity is important for this dataset. The results of the BC'17 dataset are illustrated in Figure 2.

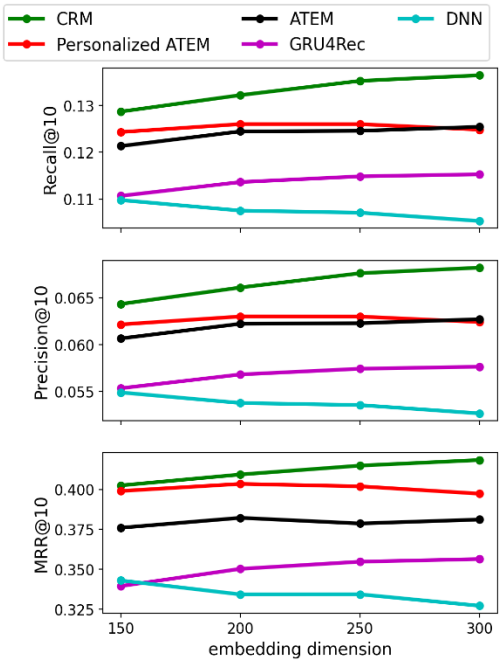

Fig. 2. The results on the BC'17 dataset

We also ran experiments using another real dataset, IJCAI'16, to demonstrate the applicability of CRM. For this dataset, we apportion the data into training and test sets, with a 90-10 split. The loss function is also optimized by Adam optimizer with a batch size of 100 and a learning rate of 0.001 in this dataset. The embedding dimensions are set to $\{60,90,120,150\}$. For this dataset, a hyperparameter λ is set to 0.5. The results of this dataset are illustrated in Figure 3.

For both datasets, DNN showed the lowest

performance. This seems to be due to the use of average pooling. In the case of ATEM, the attention layer evaluates the importance of the merchants, while DNN, and GRU4Rec apply the same significance to the merchants in a transactional context. Personalized ATEM is a personalization algorithm that includes a customer vector in the ATEM architecture. Benefiting from joint probability and more efficient representation of customers which is learned by the *Customer* network, the CRM showed the best performance.

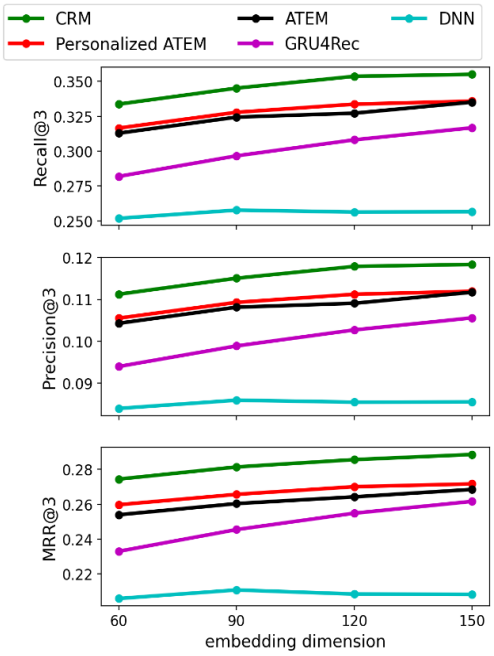

Fig. 3. The results on the IJCAI'16 dataset

CONCLUSIONS

The proposed model, CRM, showed the best performance. Given a customer and a transaction, the *Merchant* network can obtain the probability distribution of the next item. In contrast, the *Customer* network can obtain a customer probability related to transactional context. The combination of the *Merchant* and *Customer* networks can represent the joint probability of a customer and the next merchant. With the help of the *Customer* network, we can obtain more efficient customer representations that contain information about their purchases. The learned transactional context vector is also more efficient than ATEM one because it considers both customer and merchant characteristics. Consequently, significant improvements have been made.

REFERENCES

[1] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton, "Restricted Boltzmann Machines for Collaborative Filtering," *ICML '07*, pp. 791-798, New York, NY, US, Jun. 2007

[2] Diederik P. Kingma and Jimmy Ba, "A Method for Stochastic Optimization," arXiv:1412.6980, 2014

[3] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio, "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches," arXiv:1409.1259, 2014

[4] Hao Wang, Naiyan Wang, and Dit-Yan Yeung, "Collaborative Deep Learning for Recommender Systems," *KDD '15*, pp. 1235-1244, Sydney, Australia, Aug. 2015

[5] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie, "AutoRec: Autoencoders Meet Collaborative Filtering," *WWW '15*, pp. 111-112, New York, US, May 2015

[6] Balazs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk, "Session-based Recommendations with Recurrent Neural Networks", arXiv:1511.06939, 2016

[7] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Teiniu Tan, "A Dynamic Recurrent Model for Next Basket Recommendation," *SIGIR '16*, pp. 729-732, Pisa, Italy, Jul. 2016

[8] Paul Covington, Jay Adams, and Emre Sargin, "Deep Neural Networks for

YouTube Recommendations," *RecSys '16*, pp. 191-198, New York, US, Sep. 2016
[9] Byung-Ik Ahn, Ku-Imm Jung, and Hae-Lim Choi, "A Study on Recommendation Systems based on User multi-attribute attitude models and Collaborative filtering Algorithm," *Smart Media Journal*, Vol. 5, No. 2, pp. 84-89, 2016
[10] Ting Bai, Ji-Rong Wen, Jun Zhang, and Wayne Xin Zhao, "A Neural Collaborative Filtering Model with Interaction-based Neighborhood," *CIKM '17*, pp. 495-503, New York, US, Nov. 2017
[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua, "Neural Collaborative Filtering," *WWW '17*, pp. 173-182, Republic and Canton of Geneva, Switzerland, Apr. 2017
[12] Libo Zhang, Tiejian Luo, Fei Zhang, and Yanjun Wu, "A Recommendation Model Based on Deep Neural Network," *IEEE Access*, vol. 6, pp. 9454-9463, Jan. 2018
[13] Ting Bai, Jian-Yun Nie, Wayne Xin Zhao, Yutao Zhu, Pan Du, and Ji-Rong Wen, "An Attribute-aware Neural Attentive Model for Next Basket Recommendation," *SIGIR '18*, pp. 1201-1204, New York, US, Jun. 2018
[14] Shoujin Wang, Liang Hu, Longbing cao, Xiaoshui Huang, Defu Lian, and Wei Liu, "Attention-based Transactional Context Embedding for Next-Item Recommendation," *AAAI '18*, pp. 2532-2539, New Orleans, USA, Feb. 2018
[15] Hee Jun Lee, Won Sok Lee, In Hyeok Choi, and Choong Kwon Lee, "Sequence-Based Travel Route Recommendation Systems Using Deep Learning - A Case of Jeju Island -," *Smart Media Journal*, Vol. 9, No. 1, pp.45-50, 2020
[16] Hyoung Suk Kim, Jong Hyuck Lee, and Hyun Dong Lee, "Development of Personalized Clothing Recommendation Service Based on Artificial Intelligence," *Smart Media Journal*, Vol. 10, No. 1, pp. 116-123, 2021
[17] Gabriel de Souza Pereira Moreira, Sara Rabhi, Jeong Min Lee, Ronay Ak, and Even Oldridge, "Transformers4Rec: Bridging the Gap between NLP and Sequential/Session-Based Recommendation," *RecSys '21*, pp. 143-153, New York, US, Sep. 2021
[18] Ahmed Rashed, Shereen Elsayed, and Lars Schmidt-Thieme, "Context and Attribute-Aware Sequential Recommendation via Cross-Attention," *RecSys '22*, pp. 71-80, New York, US, Sep. 2022

Authors

Bayartsetseg Kalina

She received a B.S. degree in Mathematical Modeling of Economics from National University of Mongolia, Mongolia and a M.S. degree in Computer Science from Inha University, Korea. She is currently pursuing a Ph.D. degree in Computer Science from Inha University, Korea. Her research interests are deep learning, reinforcement learning, time series analysis, financial mathematics, and portfolio management.

Ju-Hong Lee

He received B.S. and M.S. degree in Computer Engineering from Seoul National University and PhD degree in Computer Science from Korea Advanced Institute of Science and Technology in Korea. His current research interest is Financial Engineering using Machine Learning. He is currently a CEO of a venture company, Qhedge Co. Ltd