Time-Aware Multibehavior Contrastive Learning for Social Recommendation

Chuyuan Wei, *Member, IEEE*, Chuanhao Hu, Chang-Dong Wang, *Senior Member, IEEE*, and Shuqiang Huang

***Abstract*—The social relationships among users can be effectively represented using graph structures, which has led to increasing interests in utilizing graph neural networks (GNNs) for social recommendation. However, there are still some inevitable issues in the existing methods: 1) The problem of sparse supervision signals in the GNN-based recommendation models has not been well addressed. 2) The existing social recommendation methods often neglect the guiding effect of the auxiliary behaviors on the target behaviors, where only the single target behavior data are used for model training. 3) In the GNN-based social recommendation algorithms, the dynamics of recommendations are rarely considered. To address these issues, this article proposes a time-aware multibehavior contrastive learning framework. To achieve better-personalized recommendation, we perform representation learning from multiview perspectives, incorporating temporal information and multibehavior interactions into the social recommendation. A time-aware GNN is then designed to model the dynamic dependency relationships between users and items, by which the dynamics of recommendations can be enhanced. Meanwhile, we propose a multibehavior contrastive learning framework to rationalize the use of multibehavioral data and address the problem of sparse supervision signals. Extensive experiments on three real-world datasets further validate the superiority of our method, where the maximum improvement can reach 6.14% in terms of NDCG@5.**

***Index Terms*—Contrastive learning, graph neural network (GNN), multibehavior learning, social recommendation, temporal information.**

Manuscript received 31 August 2023; revised 12 November 2023; accepted 14 December 2023. Date of publication 9 January 2024; date of current version 4 April 2024. This work was supported in part by National Key Research and Development Program "Industrial Software" key special project "Collaborative Optimization and Dynamic Game Decision Making of Parts Supply Chain Product Service Life Cycle Process" under Grant 2022YFB3305602, in part by the National Natural Science Foundation of China under Grant 62276277 and Grant 62272198, in part by the Humanities and Social Sciences Planning Fund of the Ministry of Education under Grant 22YJAZH110, and in part by the Guangdong Provincial Key Laboratory of Intellectual Property and Big Data under Grant 2018B030322016. Paper no. TII-23-3342. *(Corresponding author: Shuqiang Huang.)*

Chuyuan Wei is with the School of Electrical and Information Engineering, Beijing University of Civil Engineering and Architecture, Beijing 100044, China (e-mail: weichuyuan@bucea.edu.cn).

Chuanhao Hu is with the School of Mechanical-Electronic and Vehicle Engineering, Beijing University of Civil Engineering and Architecture, Beijing 100044, China (e-mail: huchuanhao@stu.bucea.edu.cn).

Chang-Dong Wang is with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510275, China, and also with the Guangdong Provincial Key Laboratory of Intellectual Property and Big Data, Guangzhou 510275, China (e-mail: changdongwang@hotmail.com).

Shuqiang Huang is with the College of Cyber Security of Jinan University, Jinan University, Guangzhou 510632, China, and also with the Guangdong Key Laboratory of Data Security and Privacy Preserving, Guangzhou 510632, China (e-mail: hsq@jnu.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2023.3345474.

Digital Object Identifier 10.1109/TII.2023.3345474

I. Introduction

According to the user's social relationship, it is often more accurate to obtain the user's preferences, making the social recommendation receive extensive attention in recent years. Generally, the traditional social recommendation mainly focuses on methods based on matrix factorization [1], [2]. However, due to the diversity and complexity of data forms and the sparseness of data content, the traditional social recommendation systems are no longer capable of providing personalized recommendations.

On account of the excellent performance of neural network technology, many researchers have attempted to utilize these techniques for social recommendation. Especially in some studies, the attention mechanism is used for social relationship modeling. For example, Chen et al. introduced [3] the attention mechanism and designed the adaptive transfer neural network EATNN to capture the mutual relationship between different users. In Diffnet++ [4], multilevel attention network was designed to learn user representations in social and collaborative domains. Meanwhile, the social recommendation methods based on graph neural networks (GNNs) have gained widespread attention due to the fact that users' social relationships can be clearly stored in graph structures and the excellent performance of GNNs in graph representation learning can be achieved. In SR-HGNN [5], global social context information was extracted through a hierarchical GNN based on a mutual information learning paradigm between low-level user embeddings and high-level global representations. Song et al. [6] used dynamic graph attention neural networks to model contextually relevant social influences as well as dynamic user behaviors. Although social recommendation models based on GNNs have achieved encouraging performances, the inherent supervisory signal sparsity in GNN is still not well addressed.

Furthermore, some studies [5], [7] point out that most of the current social recommendation methods are applicable to modeling a single type of interaction between users and items, but the user–item interaction behaviors in real scenarios are often diverse and time-sensitive. Given the variety of behaviors [8], the recommendation models must be capable of modeling user's preferences from various behaviors, and recommendations must be updated over time. Besides, considering the variety of user behaviors, more and more researches on multibehavior recommendations are made [9], [10]. However, most multibehavior recommendation systems fail to consider the guiding effect of auxiliary behaviors to target behaviors (e.g., purchase behaviors), where only the target behavior data are used to train the model. Accordingly, in the recent research [9], it has shown that the auxiliary behaviors are extremely useful in modeling user behaviors. The timeliness of user behaviors has been extensively studied in sequential recommendation, whereas few works consider the dynamics of social recommendations in the GNN-based recommendation algorithms. Some recent studies [6], [10] have taken time series information into account in social recommendation, but they are all limited to modeling a single type of user–item interaction, making it difficult to achieve optimal model performance.

To address the aforementioned issues, this article proposes a time-aware multibehavior contrastive learning framework (TMBCL) for social recommendation. In contrast to the traditional social approaches that only consider user–item interactions and user social relationships, our model takes item dependencies into account as well, by which more semantic information can be captured. To be specific, three essential components, i.e., the user social network, item contact network, and user–item multibehavior interaction graph, are designed. We can then extract user-level and item-level subgraph structures, respectively, from the user social network and item contact network, and perform recursive embedding propagation of user–item information and interaction times through a time-aware GNN, which further models the dynamic dependencies between different behaviors. In addition, a multibehavior contrastive learning module is devised to address the problem of sparse supervised signals.

The main contributions of this article are as follows.

1) We emphasize the timeliness of user behaviors, taking temporal data as input and proposing a time-aware GNN to model dynamic behavioral dependencies.
2) We design a multibehavior contrastive learning framework to capture the fine-grained differences between different types of user behaviors, considering the guidance of auxiliary behaviors toward the target behaviors.
3) Extensive experiments consisting of comparison experiments and ablation study on three real-world datasets are conducted to demonstrate the effectiveness as well as the necessity of our model.

The rest of this article is organized as follows. We provide a brief overview of related work in Section II, followed by a statement of the problem definition and our proposed method in Sections III and IV. Experimental comparisons and analysis are presented in Section V. Finally, Section VI concludes this article.

II. Related Work

A. GNN-Based Recommendation Models

GNNs have emerged as a powerful paradigm in recommendation systems, and they aim to capture intricate dependencies and interactions among nodes within graphs, making them particularly suitable for modeling complex user–item relationships. KGAT [11] leverages the knowledge graph as auxiliary information and employs graph attention network (GAT) to capture higher-order relationships through recursive propagation. NGCF [12] integrates user–item interactions into embeddings using graph convolutional network (GCN), explicitly considering high-order connectivity between users and items. TAGNN [13] enhances session-based recommendation by dynamically activating user interests based on different target items, enabling the generation of diverse representation vectors and enhancing model expressiveness.

However, in the case of GCN, feature transformations and nonlinear activation operations do not directly benefit collaborative filtering (CF). This concept is demonstrated by He et al. [14] through ablation experiments, motivating the proposal of LightGCN as a simplified version of GCN. Subsequently, numerous researchers have extended the lightweight GCN model, combining it with advanced deep learning techniques to design more effective recommendation methods. For instance, SR-HGNN [15] employs a hybrid-order gated GNN to capture intricate dependencies, effectively mitigating the oversmoothing problem. LCFN [16] merges 1-D graph convolution with a low-pass collaborative filter (LCF), achieving efficient and effective feature extraction. MixGCF [17] combines the user–item graph structure with the aggregation process of GNNs and employs leap mixing techniques to generate challenging negative samples.

B. Social Recommendation

Social recommendation is a type of recommendation method based on user social relationships, with the core idea that user decisions are influenced by their surrounding friends. SoRec [2] and TrustMF [1] are two well-known traditional social recommendation approaches. SoRec factorizes potential users' feature matrix by rating and social relationship factorization. TrustMF decomposes the social trust network into trust space and trustworthy space, which simulates the interactions between users by factorizing the social trust network and maps users to two low-dimensional spaces. Later, SEREC was proposed by Wang et al. [18], which incorporated the concept of social exposure into the matrix factorization model.

Recent social recommendation works have focused on combining themselves with advanced neural network techniques. Since user social relations can be explicitly described by graphs, social recommendation based on GNN has gained more popularity. Fan et al. [19] first used GNNs for the social recommendation, proposing GraphRec to model social relationships. Song et al. [6] used a graph attention neural network to model user-related social relations, which are then combined with a recurrent neural network to learn users' current dynamic interests. Then, Diffnet [20] and its variant Diffnet++ [4] model the recursive social diffusion process of users to capture the implicit higher-order user relationships. S4Rec [36] is a social recommendation framework that integrates information from both semantic and structural views to enhance performance, where a deep graph model and a wide attentive SVD model are utilized for rating prediction. KCGN [7] uses multibehavior modeling for the social recommendation, where the relationship-aware GNN is employed to capture social relationships containing different behavioral dependencies. Although graph-based social recommendation methods have achieved promising performances in the field of social recommendation, there are still limited researches on handling temporal data.

C. Contrastive Learning for Recommendation

As a self-supervised learning method, contrastive learning has achieved remarkable performances not only in the fields of natural language processing [22] and computer vision [23], [24] but also in the field of recommendation systems. Researchers have applied contrastive learning to various recommendation tasks, such as [25], where the authors proposed a self-supervised collaborative filtering framework (SelfCF) to solve the general recommendation task. KGCL [26] combines GNN with contrastive learning and uses knowledge graphs as auxiliary information.

The typical graph contrastive learning model, similar to SelfCF, generates new contrastive views using graph augmentation techniques and then performs contrastive learning between the views, which would, however, miss the original information [27]. Therefore, some subsequent studies no longer use graph augmentation techniques to generate contrastive views. For example, Zou et al. [28] constructed local views and global views as contrastive views based on user interaction behaviors, and proposed a multilevel cross-view contrastive learning framework (MCCLK) to address KG-Aware recommendation tasks. Similarly, Ma et al. [29] constructed bundle views as comparison views, and designed CrossCBR to use the contrastive learning mechanism for the bundle recommendation task. Inspired by the existing contrastive learning recommendation models, we propose a novel multibehavior contrastive learning paradigm for social recommendation tasks.

III. Problem Formulation

Given that $\mathcal{U} = \{u_1, u_2, \ldots, u_m\}$ and $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ denote the set of users and items, respectively, we define the user–item matrix $\mathcal{Y} \in |\mathcal{U}| \times |\mathcal{V}|$ to represent the different behaviors of users toward items. In the matrix $\mathcal{Y}$, the element $y_{i,j}$ represents different interactions between user u_i and item v_j. At the same time, we also define the user–user matrix $\mathcal{X} \in |\mathcal{U}| \times |\mathcal{U}|$ to represent the relationship between users, similar for the item–item matrix $\mathcal{Z} \in |\mathcal{V}| \times |\mathcal{V}|$. In addition, we construct the user social network $\mathcal{G}_u$, the item contact network $\mathcal{G}_v$ as well as the user–item interaction graph $\mathcal{G}_{uv}$ on the basis of the above matrix, and we store the graph in the form of the triple $\{h, e, t\}$, where h and t are graph nodes, e is the relationship between node h and t. Specifically, we further define the relevant graph-structured data as follows.

User Social Network: In order to capture the social relationship of users as auxiliary information, we define the user social network $\mathcal{G}_u = \{(u_i, e_{i,j}, u_j) \mid u_i, u_j \in \mathcal{U}\}$, where $e_{i,j}$ represents the social relationships, and $e_{i,j} = 1$ when there is a relationship between user u_i and u_j.

Item Contact Network: Similarly, to capture item dependencies as auxiliary information, we define the item contact network $G_v = \{((v_i, e_{i,j}, v_j) \mid v_i, v_j \in \mathcal{V}\}$, where $e_{i,j}$ represents the connection between items v_i and v_j. And we have $e_{i,j} = 1$ when both items have the same interaction behavior with the same user or similar attributes (e.g., color, type, etc.).

User–Item Multibehavior Interaction Graph: On the basis of the user–item matrix $\mathcal{Y}$, we define the user–item multi–behavior graph $\mathcal{G}_{uv} = \{((u_i, e_{i,j}, v_j) \mid u_i \in \mathcal{U}, v_j \in \mathcal{V}\}$, where $e_{i,j}$ represents the interaction type of the user u_i with item v_j, and $e_{i,j} = k$ when the user has the kth interaction with a certain item. In particular, to capture the dynamic preferences of the user, we extract the timestamp information of each interaction behavior. Meantime, a time vector T of the same shape is defined to store the information.

Task Statement: Given the user–user matrix $\mathcal{X}$, the user–item matrix $\mathcal{Y}$, the item–item matrix $\mathcal{Z}$, and the user–item interaction time data T, our social recommendation task is to capture dynamic dependencies and accurately predict items that users may interact with.

IV. Proposed Method

Fig. 1 depicts the specific architecture of TMBCL, where the user–item multibehavior graph is utilized as the main view, and the user social network and item contact network serve as the auxiliary views. Specifically, the model consists of the following modules.

1) Main View Representation Learning Module: In this module, user–item multibehavior interactions and time information are taken as input. First, the time information is encoded, and then a time-aware GNN is employed to learn dynamic multibehavior dependencies.
2) Auxiliary View Representation Learning Module: There are two such modules in the model, aiming to learn latent information from the user social network and item contact network. We extract connected subgraphs from the auxiliary views, and compute the cosine similarity between users and between items as the semantic similarity. Subsequently, we utilize subgraph GCN to capture the topological relationships within the graphs.
3) Multibehavior Contrastive Learning Module: In this module, the embeddings obtained from module 1) are fused and subjected to contrastive learning between positive and negative behaviors. This aims to capture fine-grained differences between different types of user behaviors through contrastive learning.
4) Multitask Optimization Module: Multitask joint optimization is executed to effectively leverage information and interdependencies among modules, ultimately enhancing the overall performance. In the following, we will provide a detailed introduction of the above modules.

A. Main View Representation Learning

To capture the dynamic behavioral dependencies, we use temporal information with the user–item multibehavior data as input to recursively learn the representation of the main view

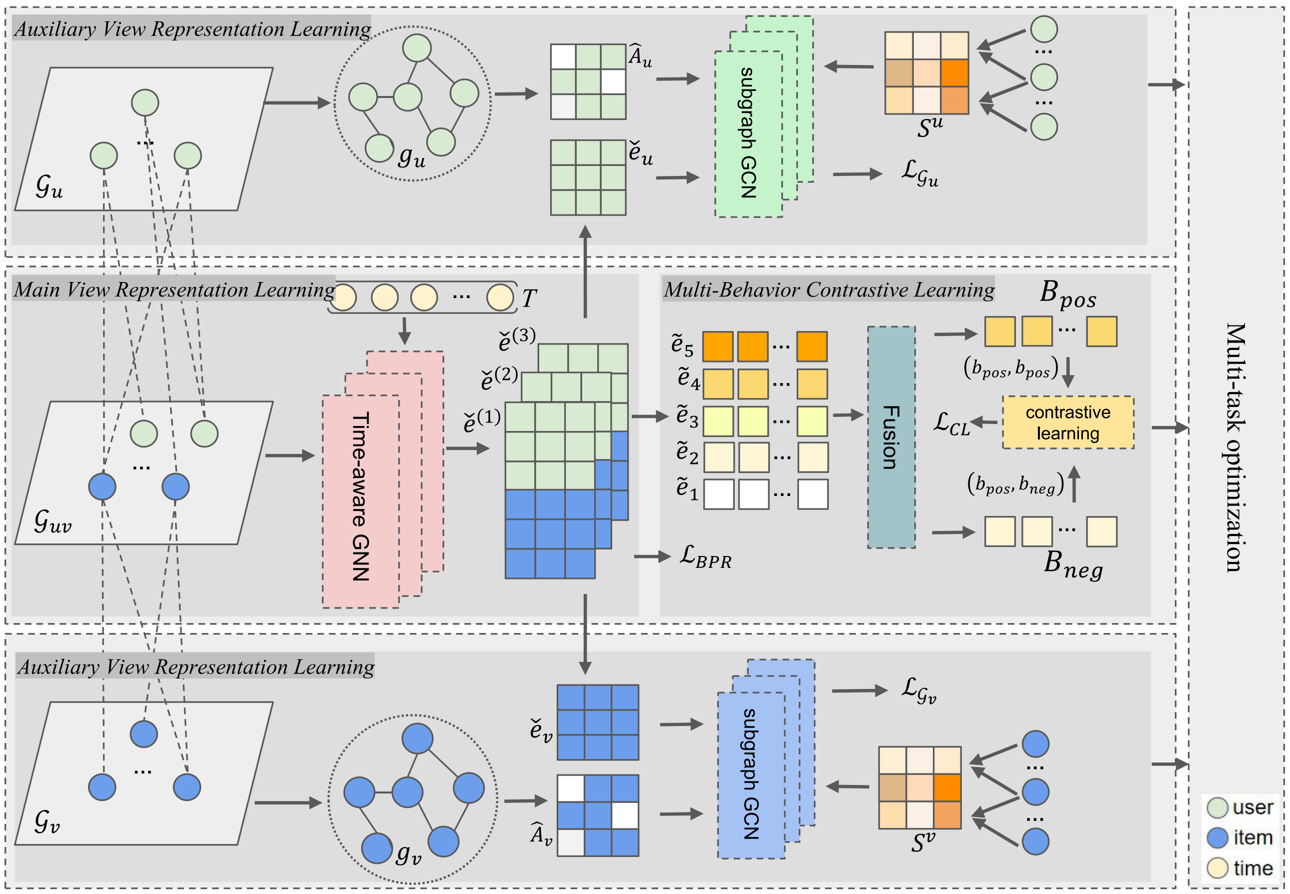


Fig. 1. Framework of TMBCL. From top to bottom, each layer corresponds to the learning of different views, and different learning objectives are set for each module. Furthermore, to balance different learning tasks, we have designed a multitask optimization module. The bottom-right corner of the figure explains the significance of the different colored circular nodes.

using the time-aware GNN. Specifically, we first normalize the raw time data $t_k^{i,j}$ corresponding to the kth behavioral interaction between user u_i and item v_j. Inspired by the temporal information modeling techniques [7], [30], we use the positional encoding mechanism in transformer architecture to map the raw time data to the temporal embedding $p_k^{i,j}$:

$$p_{k,2n}^{i,j} = \sin \frac{f\left(t_k^{i,j}\right)}{10000^{\frac{2n}{d}}}$$
$$p_{k,2n+1}^{i,j} = \cos \frac{f\left(t_k^{i,j}\right)}{10000^{\frac{2n+1}{d}}} \tag{1}$$

where $f(\cdot)$ is the standardization function, $(2n)$ and $(2n+1)$ are the temporal embedding position indices, which are the even and odd position indices, respectively. By combining the index of the time embedding with parity, cyclic information can be introduced, as the alternating changes in parity can simulate the periodic variations in sequential data. In addition, d is the potential dimension, and k is the behavior type number. Following that, we map $p_k^{i,j}$ and user–item multibehavior data to the same space using a fully connected layer, construct the multibehavior messages of users and items with temporal information, and follow the way of LightGCN [14] to make message propagation in the user–item multibehavior graph. Note that the temporal information $p_k^{i,j}$ is considered in our model, which is different from [14]. Specifically, the information propagation under the kth behavior of the $(l+1)$ layer is expressed as

$$e_{k,u_i}^{(l+1)} = \sum_{j\in\mathcal{N}_{k,u_i}} \frac{1}{\sqrt{|\mathcal{N}_{k,u_i}|\,|\mathcal{N}_{k,v_j}|}} \left(e_{k,v_j}^{(l)} \odot p_k^{i,j}\right)$$
$$e_{k,v_j}^{(l+1)} = \sum_{i\in\mathcal{N}_{k,v_j}} \frac{1}{\sqrt{|\mathcal{N}_{k,u_i}|\,|\mathcal{N}_{k,v_j}|}} \left(e_{k,u_i}^{(l)} \odot p_k^{i,j}\right) \tag{2}$$

where $\odot$ is element-wise multiplication, and $\mathcal{N}_{k,u_i}$ and $\mathcal{N}_{k,v_j}$ are the number of neighbors with the kth interaction behavior with the target node. $e_{k,u_i}^{(l)}$ is the user embedding under the kth behavior in the lth layer, and $e_{k,v_j}^{(l)}$ is the item embedding under the kth behavior in the lth layer. In particular, $e_{k,u_i}^{(0)}$ and $e_{k,v_j}^{(0)}$ are the randomly initialized user/item embedding, respectively. Under the kth behavior, the user/item embeddings of each layer learned recursively are concatenated to obtain embeddings containing different behavioral dependencies

$$\tilde{e}_{k,u_i} = e_{k,u_i}^{(0)}||e_{k,u_i}^{(1)}||\ldots||e_{k,u_i}^{(L)}$$
$$\tilde{e}_{k,v_j} = e_{k,v_j}^{(0)}||e_{k,v_j}^{(1)}||\ldots||e_{k,v_j}^{(L)} \tag{3}$$

where L is the maximum number of layers, $||$ is the concatenation operation, $\tilde{e}_{k,u_i}$ and $\tilde{e}_{k,v_j}$ are the user/item embeddings under the kth behavior. Subsequently, we use the adaptive weight matrix $W_u \in \mathbb{R}^{m\times K}$, $W_v \in \mathbb{R}^{n\times K}$ to distinguish the importance of embeddings with different behaviors, and then sum the user/item embeddings under K behaviors. Thus, we can obtain the graph-level representations $\check{e}_{u_i}$ and $\check{e}_{v_j}$

$$\check{e}_{u_i} = \sum_{k=1}^{K} W_u \widetilde{e}_{k,u_i}$$
$$\check{e}_{v_j} = \sum_{k=1}^{K} W_v \widetilde{e}_{k,v_j}. \tag{4}$$

B. Auxiliary View Representation Learning

The basic operation of GNNs is the domain aggregation of global nodes, which can be effective in learning graph representations. However, it is prone to oversmoothing problems, in addition to the huge memory and computational cost of aggregating large-scale data. To address the aforementioned issues, we extract subgraphs from auxiliary views and use subgraph convolutional networks [31] to learn topological information in the user social network $\mathcal{G}_u$ and item contact network $\mathcal{G}_v$, which reduces the computational expenditure of the auxiliary views. At the same time, considering the connections between users, we construct the similarity matrix S^u. The matrix values are calculated as follows:

$$s^u_{i,j} = \begin{cases} \frac{(e_{u_i})^T e_{u_j}}{\left\|(e_{u_i})^T\right\| \|e_{u_j}\|}, & \text{If } u_i \text{ is linked to } u_j \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where $\|\cdot\|$ is L2 regularization, e_{u_i} and e_{u_j} are the embeddings learned by users u_i and u_j on the main view $\mathcal{G}_{uv}$, and when there is a connection between the two users in auxiliary view $\mathcal{G}_u$, we compute their cosine similarity as semantic similarity comprised in the similarity matrix $S^u \in \mathbb{R}^{m\times m}$. We then normalize S^u to prevent gradient vanishing and gradient explosion problems

$$\bar{S}^u = (D)^{-\frac{1}{2}} S^u (D)^{-\frac{1}{2}} \tag{6}$$

where $D \in |U| \times |U|$ is the pairwise degree matrix. Subsequently, we recursively propagate the user/item embeddings in the auxiliary view, and we aggregate the similarity to the user embeddings in the propagation process as follows:

$$e^{(l+1)}_{u_i} = \sum_{j\in\mathcal{N}_{u_i}} \bar{s}^u_{i,j} e^{(l)}_{u_j} \tag{7}$$

where $\mathcal{N}_{u_i}$ is the neighbor nodes and $\bar{s}^u_{i,j}$ is the normalized similarity value between u_i and u_j. We then sum up the user representations on each layer to obtain the local semantic representation $E_{\mathcal{G}_{u_i}} \in \mathbb{R}^{m\times d}$

$$E_{\mathcal{G}_{u_i}} = e^{(0)}_{u_i} + e^{(1)}_{u_i} + \cdots + e^{(M)}_{u_i} \tag{8}$$

where the superscript d is the hidden layer dimension, and M is the maximum number of layers. To capture the local topological information of user social network $\mathcal{G}_u$, we extract the connected subgraph g_u from $\mathcal{G}_u$ and design the adjacency matrix $\hat{A}_u \in \mathbb{R}^{s\times m}$

$$\hat{A}_u = \begin{cases} 1, & \text{If } u_i \in g_u \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where the superscript s is the number of connected subgraphs. Then, based on the adjacency matrix $\hat{A}_u$ and the user embedding $E_{g_{u_i}}$, we can obtain the subgraph-level representation $E_{g_u} \in \mathbb{R}^{s\times d}$

$$E_{g_u} = \left(\hat{A}_u E_{g_{u_i}}\right) / N_{g_u}. \tag{10}$$

Especially, nodes from subgraph g_u are selected as positive samples, while the negative samples are randomly selected from user social network $\mathcal{G}_u$. Then, we can construct the positive and negative sample pairs $(E^{\text{pos}}_{\mathcal{G}_{u_i}}, E_{g_u})$, $(E^{\text{neg}}_{\mathcal{G}_{u_i}}, E_{g_u})$. Due to the cross-entropy rationality in mutual information maximization, we design the loss function as follows:

$$\mathcal{L}_{\mathcal{G}_u} = -\frac{\lambda_u}{N_{\text{pos}} + N_{\text{neg}}} \left(\sum_{i=1}^{N_{\text{pos}}} \phi\left(E^{\text{pos}}_{\mathcal{G}_{u_i}}, E_{g_u}\right) \log\sigma\left(E^{\text{pos}}_{\mathcal{G}_{u_i}} E_{g_u}\right) + \sum_{i=1}^{N_{neg}} \phi\left(E^{neg}_{\mathcal{G}_{u_i}}, E_{g_u}\right) \log\left(1 - \sigma\left(E^{neg}_{\mathcal{G}_{u_i}} E_{g_u}\right)\right)\right) \tag{11}$$

where N_{pos} and N_{neg} are the number of positive and negative sample pairs, respectively. $\phi(\cdot)$ is the indicator function, and $\sigma(\cdot)$ is the activation function. Similarly to $\mathcal{L}_{\mathcal{G}_u}$, $\mathcal{L}_{\mathcal{G}_v}$ in the item contact network can also be calculated. Therefore, we can obtain the loss function $\mathcal{L}_{\mathcal{G}_{AUX}}$ for the auxiliary view by combining $\mathcal{L}_{\mathcal{G}_u}$ with $\mathcal{L}_{\mathcal{G}_v}$ through a balanced hyperparameter α

$$\mathcal{L}_{\text{AUX}} = (1-\alpha)\,\mathcal{L}_{\mathcal{G}_u} + \alpha\mathcal{L}_{\mathcal{G}_v}. \tag{12}$$

C. Multibehavior Contrastive Learning

To address the sparsity issues of multibehavior interaction data and supervision signals, we design a fusion multibehavior contrastive learning framework. Considering the guidance of the auxiliary behavior to the target behavior, as well as the commonality and difference between different behaviors, we selectively fuse the embeddings of different behaviors to generate positive behavior embedding B_{pos} and negative behavior embedding B_{neg}. Specifically, we use the division of [7] to classify the interaction behavior data into five behavior types, namely negative, below average, average, above average, and positive behavior. Then, we fuse the fourth (i.e., above average) and fifth (i.e., positive) behavior representations as the positive behavior embedding B_{pos}, and the first three behavior representations as the negative behavior embedding B_{neg}. In order to map the embeddings of different behaviors to the same dimension, we use a multilayer perceptron to fuse the user/item embeddings of different behaviors, whose formulation is as follows:

$$B^u_{neg} = \varphi\left(\tilde{e}_{1,u} \|\tilde{e}_{2,u}\| \,\tilde{e}_{3,u}\right)$$
$$B^u_{pos} = \varphi\left(\tilde{e}_{4,u} \|\tilde{e}_{5,u}\right)$$

$$B^{v}_{neg} = \varphi\left(\tilde{e}_{1,v} \|\tilde{e}_{2,v}\| \, \tilde{e}_{3,v}\right)$$
$$B^{v}_{pos} = \varphi\left(\tilde{e}_{4,v} \|\tilde{e}_{5,v}\right) \tag{13}$$

where $\varphi(\cdot)$ is a multilayer perceptron with hidden layers, $\tilde{e}_{k,u}$ $\tilde{e}_{k,v}$ is the multi-behavior representation calculated by (3). Then, we select the positive sample pair $(b^{u_i}_{pos}, b^{u_j}_{pos})$, $(b^{v_o}_{pos}, b^{v_p}_{pos})$, the negative sample pair $(b^{u_i}_{pos}, b^{u_k}_{neg})$, $(b^{v_o}_{pos}, b^{v_q}_{neg})$ from B_{pos} and B_{neg}, and use InfoNCE [24] as the contrastive learning loss function, which is constructed by

$$\mathcal{L}_{CL} = -(1-\varepsilon)\log\frac{\exp\left(\psi\left(b^{u_i}_{pos}, b^{u_j}_{pos}\right)/\tau\right)}{\sum_{b\in B}\exp\left(\psi\left(b^{u_i}_{pos}, b^{u_k}_{neg}\right)/\tau\right)} - \varepsilon\,\log\frac{\exp\left(\psi\left(b^{v_o}_{pos}, b^{v_p}_{pos}\right)/\tau\right)}{\sum_{b\in B}\exp\left(\psi\left(b^{v_o}_{pos}, b^{v_q}_{neg}\right)/\tau\right)} \tag{14}$$

where τ is the temperature parameter, $\psi(\cdot)$ is the cosine similarity calculation function, and ε is a tradeoff hyperparameter.

D. Multitask Optimization

To balance the tasks of different views and optimize the model, we use a multitask optimization strategy to combine the recommendation task with the self-supervised task. Specifically, we adopt the Bayesian Personalized Ranking (BPR) [33] loss as the loss function for the recommendation task

$$\mathcal{L}_{\text{BPR}} = \sum_{(i,j,j^-)\in O} -\ln\sigma\left(\hat{\mathbf{y}}_{i,j} - \hat{\mathbf{y}}_{i,j^-}\right) \tag{15}$$

where $O = \{(i,j,j^-) \mid (i,j)\in O^+, (i,j^-)\in O^-\}$ is the pairwise training dataset made up of observed interactions O^+ and unobserved interactions O^-, and $\sigma(\cdot)$ is the sigmoid activation function. Following that, we jointly consider the pairwise BPR loss function $\mathcal{L}_{\text{BPR}}$, the auxiliary view loss $\mathcal{L}_{\mathcal{G}_{\text{AUX}}}$, and the multi-behavior contrastive learning loss function $\mathcal{L}_{CL}$ to minimize the following objective function:

$$\mathcal{L}_{\text{TMBCL}} = \mathcal{L}_{\text{BPR}} + \lambda_1\mathcal{L}_{\mathcal{G}_{\text{AUX}}} + \lambda_2\mathcal{L}_{CL} + \lambda_3\|\Theta\|^2 \tag{16}$$

where Θ are trainable model parameters, λ_1 and λ_2 are the hyperparameters to balance auxiliary view loss and contrastive learning loss, and λ_3 is used to control the strength of L2 regularization for overfitting alleviation.

E. Time Complexity Analysis

For clarity, we summarize the training process of TMBCL in Algorithm 1, where L and M are the numbers of convolutional layers for the main view and auxiliary view, respectively, and K is the number of behaviors. Starting from initializing the model parameters and embeddings (line 1), we then construct three graphs based on the input data (line 2). Subsequently, we perform N iterations to iteratively compute user/item embeddings $e^{(l)}_{k,u_i}$ and $e^{(l)}_{k,v_j}$ under the kth behavior layer by layer (line 7). In the lth layer, for a given user–item pair (u_i, v_j), we aggregate the corresponding temporal vectors $p^{i,j}_k$ and neighborhood information to achieve the output (line 8). Next, we derive distinct behavior representations $\tilde{e}_{k,u_i}/\tilde{e}_{k,v_j}$ based on the learned user/item embeddings (line 10) and obtain the graph-level representation $\check{e}_{u_i}/\check{e}_{v_j}$ (line 12). Concurrently, we conduct M iterations to

Algorithm 1: Training Procedure of TMBCL.

```
Input: User-user matrix $\mathcal{X}$, user-item matrix $\mathcal{Y}$,
  item-item matrix $\mathcal{Z}$, user-item interaction time data $T$.
Output: Top-$K$ recommendations.
 1: Initialize the learnable model parameters, user
    embedding, and item embedding;
 2: Construct $\mathcal{G}_u$, $\mathcal{G}_v$, and $\mathcal{G}_{uv}$;
 3: Compute the temporal vector $p^{i,j}_k$ via (1) of $t^{i,j}_k \in T$;
 4: repeat
 5:   for $u_i, v_j \in \mathcal{Y}$ do
 6:     for $k$ to $K$ do
 7:       for $l$ to $L$ do
 8:         Compute user/item embeddings $e^{(l)}_{k,u_i}$ and $e^{(l)}_{k,v_j}$
            for different layers via (2);
 9:       end for
10:       Obtain the embedding vector for different
          behaviors $\tilde{e}_{k,u_i}$ and $\tilde{e}_{k,v_j}$ via (3);
11:     end for
12:     Obtain the graph-level representation $\check{e}_{u_i}$ and $\check{e}_{v_j}$ of
        the main view $\mathcal{G}_{uv}$ via (4);
13:     for $l$ to $M$ do
14:       Perform representation learning for auxiliary
          views via (5)–(10);
15:     end for
16:   end for
17:   Perform multi-behavior contrastive learning via (13)
      and (14);
18:   Calculate the pairwise BPR loss function $\mathcal{L}_{BPR}$ and
      auxiliary view loss functions $\mathcal{L}_{\mathcal{G}_u}$ and $\mathcal{L}_{\mathcal{G}_v}$ via (11),
      (12), and (15);
19:   Multi-task joint optimization via (16);
20:   Update model parameters;
21: until Equation (16) converges
22: return Top-$K$ recommendations.
```

learn auxiliary view information (line 14). Based on the learned user/item embeddings, we calculate the loss function for different tasks (lines 18–19), until the convergence of our multitask optimization function (line 21).

As deduced from the above, TMBCL initially takes $O((x + y + z) \times d)$ for constructing views, where x, y, and z represent the number of edges in $\mathcal{G}_u$, $\mathcal{G}_{uv}$, and $\mathcal{G}_v$, respectively. In addition, we also need $O(y \times d)$ for mapping temporal information. For the three views, our model requires $O((m + m) \times d^2)$, $O((m + n) \times d^2)$, and $O((n + n) \times d^2)$, respectively, for iterative aggregation. Due to the consideration of multibehavior modeling on the main view, learning on the main view actually demands $O((m + n \times K) \times d^2)$. Furthermore, similar to the majority of GCN methods, our model also needs $O((m + n \times K) \times d)$ for storing the learned embeddings of users and items.

V. Experiments and Analysis

A. Datasets

We evaluate TMBCL on three real-world datasets, i.e., Yelp, Epinions, and CiaoDVD, whose statistics are shown in Table I.

TABLE I
STATISTICS OF EXPERIMENTAL DATASETS

Statistics	Yelp	Epinions	CiaoDVD
#Users	43 043	18 081	17 615
#Items	66 576	251 722	16 121
#User–Item Interactions	283 512	715 821	72 665
#User–Item Interaction Density	0.00989%	0.01573%	0.02558%
#User Ties	549 451	590 641	22 484
#Social Density	0.02965%	0.18066%	0.00725%
#Item Relations	1 847 060	6 069 106	-
#Item Relation Density	0.04167%	0.09578%	-

Yelp.[1] This dataset comes from the Yelp platform and contains a large amount of businesses, reviews, and user data. The user–item interactions are segmented in the same way as [7].

Epinions.[2] This dataset is obtained from the social networking site Epinions and contains a large number of users' social relationships as well as behaviors toward different items. The rating scale goes from 1 to 5 (i.e., negative, below average, neutral, above average, positive), with each rating being considered as an individual behavioral interaction.

CiaoDVD.[3] This dataset is compiled from a wide spectrum of DVD categories. It encompasses various DVDs, along with accompanying user reviews, ratings, and social connections between users. However, it does not include item-to-item connections. We have filtered out any irrelevant user connection information from this dataset.

B. Evaluation Protocols

We choose the hit rate (HR@10) and normalized discounted cumulative gain (NDCG@10) as our evaluation metrics to compare the performance of our proposed approach against the baseline methods. The leave-one-out method is adopted to divide the training and test sets, where each positive instance is associated with 99 negative samples to ensure a fair comparison.

C. Baseline

1) *TrustMF* [1]*:* TrustMF is a classical social recommendation method based on matrix factorization.
2) *GraphRec* [19]*:* GraphRec aggregates social relationships between users from graph data in social recommendations via graph neural structures.
3) *DiffNet* [33]*:* DiffNet is a model with hierarchical influence propagation structure to simulate recursive dynamic social diffusion in social recommendation.
4) *DGRec* [6]*:* DGRec is a dynamic graph attention network approach for modeling dynamic user interests and context-dependent social influences.
5) *DANSER* [34]*:* It consists of two graphic attention networks that capture social impact by learning the representation of dual social effects.
6) *LR-GCCF* [35]*:* LR-GCCF combines linear residual graph convolution methods with graph-based CF models to better serve social recommendations.
7) *ConsisRec* [21]*:* ConsisRec addresses the social inconsistency problem by employing the relation attention mechanism to assign consistent relations.
8) *S4Rec* [36]*:* S4Rec is a social recommendation framework that utilizes a deep graph model and a wide attentive SVD model for rating prediction.
9) *KCGN* [7]*:* It uses a knowledge-aware coupled GNN to aggregate higher-order information between the users and items.
10) *MCCLK* [28]*:* MCCLK is an advanced knowledge-aware multiview contrastive learning framework.

D. Parameter Settings

The parameter settings of the baseline models follow their original setups and are fine-tuned within the suggested parameter ranges. We use PyTorch to implement the TMBCL framework and use the easy-to-use ADAM optimizer to optimize the model parameters. To ensure the convergence speed of the model and avoid instability, the initial value of the learning rate is set to $1e$-3 and dynamically adjusted in the range of $1e$-4 to $1e$-2 during the model training process. The embedding parameters are initialized with the Xavier method, and the embedding size is chosen from $\{32, 64, 128\}$. The batch size is chosen from $\{512, 1024, 2048\}$. The hyperparameters ε and α for balancing the weights of the learning tasks on the user and the item are set to 0.5 defaultly, and are adjusted according to the dataset characteristics in $\{0.1, 0.2, \ldots, 0.9\}$. Other than that, the other hyperparameters follow the settings in [7] taken from $\{1e-2, 1e-3, 1e-4, 1e-5\}$.

E. Performance Comparison

In Tables II and III, we show the experimental results of TMBCL and ten baselines, where HR@k and NDCG@k are used as evaluation metrics, and k takes values of 5, 10, and 15. In general, TMBCL achieves the best results on Yelp, Epinions, and CiaoDVD. As an exemplary matrix decomposition-based social recommendation model, TrustMF achieves good performance on Epinions while compared with the mainstream social recommendation model. However, TrustMF does not perform well on Yelp and CiaoDVD, demonstrating that the traditional matrix factorization model cannot well solve the data sparsity problem. Overall, the recommendation models based on GNN outperform traditional matrix factorization methods, since the recursive propagation paradigm can capture high-order implicit connections from explicit interactions. However, compared with the basic GNN social recommendation model, such as GraphRec, there is no significant improvement for the neural network model with graph attention mechanism. A possible reason is that simulating the user's social relationships through calculating specific weights in the mathematical paradigm is challenging.

In addition, knowledge-aware recommendation methods have achieved good results, among which KCGN and MCCLK are recommendation models with knowledge-aware capabilities,

[1][Online]. Available: https://www.yelp.com/dataset/download
[2][Online]. Available: https://www.cse.msu.edu/tangjili/datasetcode/truststudy.html
[3][Online]. Available: https://guoguibing.github.io/librec/datasets.html

TABLE II
PERFORMANCE OF TMBCL AND TEN COMPARED BASELINES ON YELP, EPINIONS, AND CIAODVD. THE BEST RESULTS IN THE BASELINE ARE BOLDED AND THE RESULTS OF OUR MODEL ARE BOLDED AND UNDERLINED. THE *IMPROVEMENT* OVER THE BEST DEEP LEARNING MODELS IS RECORDED

Datasets	Yelp			Epinions			CiaoDVD		
	NDCG@5	NDCG@10	NDCG@15	NDCG@5	NDCG@10	NDCG@15	NDCG@5	NDCG@10	NDCG@15
TrustMF	0.4451	0.4959	0.5139	0.3738	0.4179	0.4263	0.4013	0.4587	0.4720
GraphRec	0.4504	0.4943	0.5137	0.4251	0.4786	0.4901	0.4421	0.4894	0.5049
DiffNet	0.4622	0.5126	0.5329	0.3726	0.4160	0.4257	0.4224	0.4689	0.4841
DANSER	0.4624	0.5082	0.5245	0.4144	0.4627	0.4736	0.4445	0.4920	0.5057
DGRec	0.4445	0.4954	0.5141	0.3697	0.4127	0.4225	0.4185	0.4595	0.4739
LR-GCCF	0.4766	0.5189	0.5302	0.4203	0.4783	0.4875	0.4547	0.5036	0.5139
ConsisRec	0.4508	0.5016	0.5232	0.4235	0.4744	0.4762	0.4437	0.4880	0.4997
S4Rec	0.4785	0.5133	0.5339	0.4278	0.4777	0.4883	**0.4618**	0.5074	**0.5234**
KCGN	0.4876	0.5308	0.5424	**0.4284**	**0.4792**	**0.4910**	0.4580	**0.5090**	0.5208
MCCLK	**0.4877**	**0.5318**	**0.5425**	0.4272	0.4750	0.4894	0.4483	0.5034	0.5160
TMBCL	**<u>0.4996</u>**	**<u>0.5512</u>**	**<u>0.5635</u>**	**<u>0.4547</u>**	**<u>0.4969</u>**	**<u>0.5161</u>**	**<u>0.4638</u>**	**<u>0.5241</u>**	**<u>0.5398</u>**
Improvement	**2.44%**	**3.64%**	**3.89%**	**6.14%**	**3.69%**	**5.11%**	**0.43%**	**2.97%**	**3.13%**

The best results in the baseline are bolded and the results of our model are bolded and underlined. The *improvement* over the best deep learning models is recorded.

TABLE III
PERFORMANCE OF TMBCL AND TEN COMPARED BASELINES ON YELP, EPINIONS, AND CIAODVD. THE BEST RESULTS IN THE BASELINE ARE BOLDED AND THE RESULTS OF OUR MODEL ARE BOLDED AND UNDERLINED. THE *IMPROVEMENT* OVER THE BEST DEEP LEARNING MODELS IS RECORDED

Datasets	Yelp			Epinions			CiaoDVD		
	HR@5	HR@10	HR@15	HR@5	HR@10	HR@15	HR@5	HR@10	HR@15
TrustMF	0.6065	0.7562	0.8337	0.5128	0.6353	0.6932	0.5697	0.7055	0.7741
GraphRec	0.6233	0.7605	0.8342	0.5565	0.6865	0.7491	0.5993	0.7235	0.8043
DiffNet	0.6311	0.7853	0.8628	0.5122	0.6323	0.6895	0.5819	0.7216	0.7901
DANSER	0.6304	0.7740	0.8356	0.5420	0.6693	0.7297	0.5968	0.7289	0.7967
DGRec	0.6114	0.7662	0.8399	0.5077	0.6268	0.6835	0.5719	0.7035	0.7693
LR-GCCF	0.6319	0.7692	0.8421	0.5491	0.6779	0.7392	0.6011	0.7337	0.8049
ConsisRec	0.6235	0.7607	0.8349	0.5562	0.6857	0.7454	0.6005	0.7232	0.8044
S4Rec	0.6454	0.7864	0.8490	0.5687	0.7016	0.7610	**0.6180**	**0.7589**	**0.8211**
KCGN	0.6594	**0.8026**	0.8682	**0.5715**	**0.7018**	**0.7667**	0.6155	0.7522	0.8175
MCCLK	**0.6619**	0.8013	**0.8697**	0.5685	0.6963	0.7649	0.6029	0.7338	0.8091
TMBCL	**<u>0.6700</u>**	**<u>0.8181</u>**	**<u>0.8803</u>**	**<u>0.5931</u>**	**<u>0.7198</u>**	**<u>0.7884</u>**	**<u>0.6177</u>**	**<u>0.7690</u>**	**<u>0.8344</u>**
Improvement	**1.22%**	**1.93%**	**1.22%**	**3.78%**	**2.56%**	**2.83%**	**-**	**1.33%**	**1.62%**

The best results in the baseline are bolded and the results of our model are bolded and underlined. The *improvement* over the best deep learning models is recorded.

and MCCLK is a state-of-the-art multiview knowledge-aware framework. It is worth noting that both MCCLK and our model perform well on Yelp due to their capability of handling sparsity in user–item interactions, user social connections, and item relationships. Despite this, TMBCL can further well capture the data characteristics, and achieves better performances than MCCLK, whose improvements are, respectively, 2.44%, 3.64%, and 3.89% in terms of NDCG@5/10/15 on Yelp.

Moreover, KCGN performs well on all three datasets, and especially on Epinions, where the incorporation of multibehavior modeling into social recommendation plays a leading role for the excellent performance. Similar to KCGN, we also take into account different user behavior interactions in TMBCL, and use the multibehavior contrastive learning framework to learn user behavior representations, where the differences between different behaviors are expanded and the main relationships between them are learned. It is worth mentioning that our model performs better than KCGN on Epinions, whose improvements are 6.14%, 3.69%, and 5.11% in terms of NDCG@5/10/15, respectively. A reasonable explanation is that our model introduces a similarity matrix in the auxiliary view learning phase, which allows better learning of users' social relationships from rich social networks, and better extraction of item interrelationships from item contact networks. S4Rec is the best single-behavior social recommendation model that fully utilizes semantic and structural views, but its performance is still slightly inferior to KCGN and TMBCL on most metrics, further demonstrating the importance of taking multiple behaviors into consideration for recommendation.

Compared to the Yelp and Epinions datasets, the CiaoDVD dataset is richer in user–item interactions but very sparse in social relationships. This data characteristic requires social recommendation models to have the ability to learn important social relationships from sparse data. In addition, there is a lack of connectivity relationships between items in this dataset. This results in multiview models, such as MCCLK and KCGN, not performing as well as they did on the first two datasets. However, TMBCL still outperforms other methods on most of the metrics on this dataset due to its excellent ability to handle sparse data, which proves the generalizability and superiority of our approach.

TABLE IV
PERFORMANCE COMPARISON OF TMBCL AND ITS VARIANTS ON YELP AND EPINIONS

Datasets	Yelp		Epinions	
	HR@10	NDCG@10	HR@10	NDCG@10
TMBCL	0.8181	0.5512	0.7198	0.4939
TMBCL-TA	0.8021	0.5309	0.7091	0.4828
Decrease	1.96%	3.68%	1.49%	2.25%
TMBCL-SG_u	0.8006	0.5272	0.7138	0.4875
Decrease	2.14%	4.35%	0.83%	1.30%
TMBCL-SG_v	0.7918	0.5235	0.6918	0.4756
Decrease	3.21%	5.02%	3.89%	3.71%
TMBCL-SG_{uv}	0.7884	0.5212	0.6895	0.4748
Decrease	3.63%	5.44%	4.21%	3.87%
TMBCL-CL	0.8018	0.5296	0.7083	0.4764
Decrease	1.99%	3.92%	1.60%	3.54%
TMBCL-MV	0.7827	0.5169	0.6871	0.4677
Decrease	4.33%	6.22%	4.54%	5.30%

F. Ablation Study

In this part, the ablation study is conducted, where the following six comparative variants of TMBCL are designed to demonstrate the effectiveness of the multibehavior contrastive learning framework, time-aware GNN, multiview learning framework, and subgraph information aggregation.

1) TMBCL-TA: We replace time-aware GNN with GCN.
2) TMBCL-SG_u: Only use GCN to learn the representation of user social network $\mathcal{G}_u$.
3) TMBCL-SG_v: Only use GCN to learn the representation of item contact network $\mathcal{G}_v$.
4) TMBCL-SG_{uv}: Only use GCN to learn the representation of the auxiliary view.
5) TMBCL-CL: We remove the multibehavior contrastive learning module.
6) TMBCL-MV: We remove the auxiliary view representation learning module.

Table IV presents the experimental results of TMBCL and its six variants on the two datasets, where important insights into the contribution of different components to the model can be observed. Overall, TMBCL still achieves the best performance. And TMBCL-MV, which removes the multiview framework, performs the worst among all the variants, showing a decrease of 4.33%/6.22% and 4.54%/5.30% in terms of HR@10 and NDCG@10 on the two datasets, respectively. This suggests that the social relations as well as item connections are crucial for modeling user preferences. In addition, TMBCL-SG_{uv}, which removes the connected subgraph extraction and similarity computation on both user social network $\mathcal{G}_u$ and item contact network $\mathcal{G}_v$, also shows a significant performance gap compared with TMBCL, which indicates that the subgraph extraction and similarity calculation components play an important role in improving the performance of the model.

Interestingly, we observe that TMBCL-SG_u performs better than TMBCL-SG_v on both datasets, indicating that the connected subgraph extraction and similarity calculation components on the item contact network are more important than those on the user social network. This could be due to the fact that there is less data on user social interactions on Yelp and Epinions than that on item relationships. Furthermore, since the number of users on Epinions is less than one-tenth of the number of items and the data on user social interactions is less than the data on item relationships, the extraction of connected subgraphs and similarity calculation components on the user social network is less important on this dataset. The performance of TMBCL-SG_u on Epinions (with only a slight decrease of 0.83%/1.30% in terms of HR@10 and NDCG@10) also precisely confirms this hypothesis. Moreover, the significant decrease in the effectiveness of TMBCL-TA and TMBCL-CL demonstrates the importance of temporal data for recommendations and the effectiveness of the reasonably embedded contrastive learning mechanism for solving the problem of supervised signal sparsity. However, relatively speaking, the decrease of TMBCL-TA is not particularly significant, especially with a decrease of nearly 1.49%/2.25% in terms of HR@10 and NDCG@10 on Epinions, which indicates that our proposed time-aware GNN still requires further improvement and refinement.

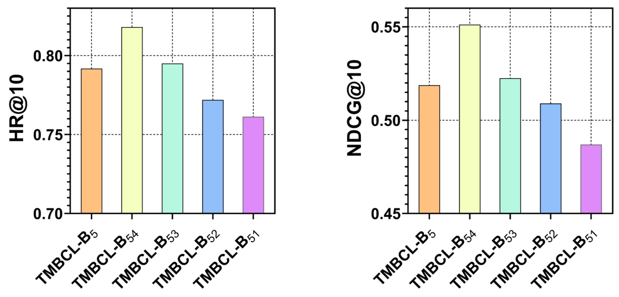


Fig. 2. Performance of different fusion schemes on Yelp.

G. Analysis of Different Behaviors Contrastive Learning

As shown in Fig. 2, in order to prove the rationality of this scheme, we designed five comparative schemes. Among them, TMBCL-B_{ij} represents the fusion of the ith behavior representation $\tilde{e}_{i,u}/\tilde{e}_{i,v}$ and the jth behavior representation $\tilde{e}_{j,u}/\tilde{e}_{j,v}$ as positive behavior, while other behaviors are fused as negative behavior. In particular, B^u_{pos}/B^v_{pos} in scheme TMBCL-B_5 is fused from $\tilde{e}_{5,u}$ /$\tilde{e}_{5,v}$ alone. TMBCL-B_{54} is our default fusion scheme as shown in (13). In Fig. 2, the performances of different fusion schemes on Yelp are reported, and TMBCL-B_{54} maintains the best performance. Through comparison with other fusion schemes, it is evident that incorporating below-average behaviors into B_{pos} for contrastive learning significantly impacts the model's performance, which highlights the importance of applying contrastive learning appropriately.

In addition, to analyze the reasons for the performance differences among different fusion schemes, we compute the alignment and uniformity of embeddings based on [37]. Alignment and uniformity are closely related to the effectiveness of recommendations, and smaller values indicate better alignment and uniformity, which often leads to superior learned

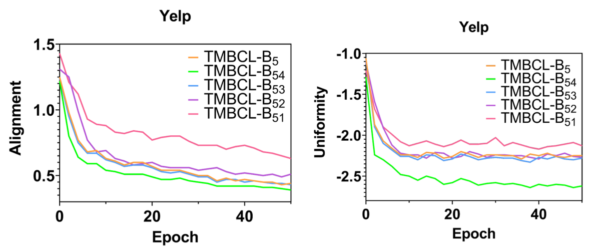


Fig. 3. Alignment and uniformity between user representations and item representations in different fusion schemes.

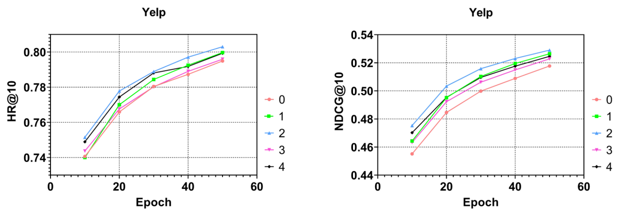


Fig. 4. Performance of time-aware GNN with different numbers of layers during the first 50 epochs.

representations. Fig. 3 shows the alignment and uniformity measures during the training process of 50 epochs. TMBCL-B_{51} exhibits the worst alignment and uniformity, followed by TMBCL-B_{52}. TMBCL-B_{53} and TMBCL-B_5 show similar alignment and uniformity during the training process. It can be observed that incorporating below-average behaviors into B_{pos} leads to worse alignment and uniformity, which validates the rationality of the fusion scheme in (3).

H. Parameter Sensitivity Study

1) Impact of the Number of Time-Aware GNN Layers: Fig. 4 demonstrates the impact of the recursive layers of the time-aware GNN on model performance using the Yelp dataset. We can observe that the model achieves the best performance in terms of both metrics when the number of convolutional layers is set to $l = 2$, outperforming the cases when $l = 0$ and $l = 1$. Furthermore, the performance of the model with $l = 1$ is better than that of $l = 0$, indicating that increasing the number of time-aware GNN layers can enhance performance. However, when $l > 2$, the model performance deteriorated, with a 3-layer convolutional time-aware GNN consistently achieving lower accuracy throughout the training process. This is attributed to the over-smoothing phenomenon that occurs with excessive layers in GNNs, resulting in decreased performance.

2) Impact of Balance Parameters: We set the balance parameters λ_1 and λ_2 in (16) in the range of $\{0.001, 0.01, 0.05, 0.1, 1\}$, and for α in (12) and ε in (14), the range of values is set to $\{0.2, 0.3, 0.4, 0.5, 0.6, 0.7\}$. As shown in Fig. 5, we observe that as λ_1 and λ_2 decrease, the performance of TMBCL gradually improves on both datasets, reaching its peak at $\lambda_1 = 0.05$ and $\lambda_2 = 0.01$, and then starts to decline. The reason is that when λ_1 and λ_2 are large, the model tends to prioritize these two loss functions and their corresponding tasks, resulting in the insufficient training on the main view learning task and decreased performance. However, the main view contains rich time information and multibehavior interaction data, which can provide better guidance for the model to learn. Therefore, when $\lambda_1 = 1$, TMBCL performs very poorly on the Yelp dataset. As λ_1 and λ_2 decrease, the main view loss function becomes more dominant, allowing the model to focus on learning from the main view and leverage the auxiliary views to improve the representation learning.

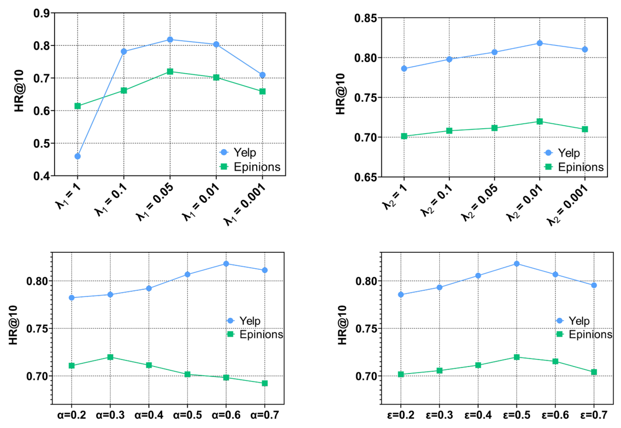


Fig. 5. Performance of different balance parameters on TMBCL.

α is a balance parameter about the auxiliary view, with larger values of α, the focus of the auxiliary view learning task becomes more and more oriented toward the item contact network, and vice versa toward the user social network. In the Yelp and Epinions datasets, the sparsity of social and item relationships is different. Consequently, the conditions for TMBCL to reach its peak performance vary on the two datasets. For instance, in the Yelp dataset, where social relationships are relatively sparse, the peak is achieved at $\alpha = 0.6$, which implies that the model works better when focusing on the learning of item relationships. This coincides with the fact that item-connected relationships in this dataset are denser and contain more information. In addition, ε functions similarly to α as a balance parameter in the contrastive learning loss function. TMBCL reaches its peak performance at $\varepsilon = 0.5$ on both datasets.

VI. Conclusion

In this article, we focused on exploring contrastive learning in social recommendation and proposed a time-aware multibehavior contrastive learning framework TMBCL. Within this framework, we conducted representation learning from multiview perspectives, and incorporate time information as well as multibehavior interactions into the social recommendation. A time-aware GNN was then specifically designed to model dynamic dependencies. In addition, we designed subgraph extraction and similarity calculation components, using subgraph GCN to capture more semantic information from auxiliary views. To address the sparsity of supervision signals, we designed a multibehavior contrastive learning framework. Extensive experiments on three public datasets demonstrated the superiority of our proposed method.

References

[1] B. Yang, Y. Lei, J. Liu, and W. Li, "Social collaborative filtering by trust," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1633–1647, Aug. 2017.

[2] H. Ma, H. Yang, M. R. Lyu, and I. King, "Sorec: Social recommendation using probabilistic matrix factorization," in *Proc. 17th ACM Conf. Inf. Knowl. Manage.*, 2008, pp. 931–940.

[3] C. Chen et al., "An efficient adaptive transfer neural network for social-aware recommendation," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 225–234.

[4] L. Wu, J. Li, P. Sun, R. Hong, Y. Ge, and M. Wang, "Diffnet : A neural influence and interest diffusion network for social recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 10, pp. 4753–4766, Oct. 2020.

[5] H. Xu, C. Huang, Y. Xu, L. Xia, H. Xing, and D. Yin, "Global context enhanced social recommendation with hierarchical graph neural networks," in *Proc. IEEE Int. Conf. Data Min.*, 2020, pp. 701–710.

[6] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, 2019, pp. 555–563.

[7] C. Huang et al., "Knowledge-aware coupled graph neural network for social recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 5, pp. 4115–4122.

[8] G. Balloccu, L. Boratto, G. Fenu, and M. Marras, "Post processing recommender systems with knowledge graphs for recency, popularity, and diversity of explanations," in *Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2022, pp. 646–656.

[9] S. Gu, X. Wang, C. Shi, and D. Xiao, "Self-supervised graph neural networks for multi-behavior recommendation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2022. pp. 2052–2058.

[10] L. Xia et al., "Knowledge-enhanced hierarchical graph transformer network for multi-behavior recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 5, pp. 4486–4493.

[11] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "Kgat: Knowledge graph attention network for recommendation," in *Proc. 28th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 950–958.

[12] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 165–174.

[13] F. Yu, Y. Zhu, Q. Liu, S. Wu, L. Wang, and T. Tan, "Tagnn: Target attentive graph neural networks for session-based recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 1921–192.

[14] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 639–648.

[15] Y.-H. Chen, L. Huang, C.-D. Wang, and J.-H. Lai, "Hybrid-order gated graph neural network for session-based recommendation," *IEEE Trans. Ind. Informat.*, vol. 18, no. 3, pp. 1458–1467, Mar. 2022.

[16] W. Yu, Z. Zhang, and Z. Qin, "Low-pass graph convolutional network for recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2022, vol. 36, no. 8, pp. 8954–8961.

[17] T. Huang et al., "Mixgcf: An improved training method for graph neural network-based recommender systems," in *Proc. 27th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2021, pp. 665–674.

[18] M. Wang, X. Zheng, Y. Yang, and K. Zhang, "Collaborative filtering with social exposure: A modular approach to social recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2018, vol. 32, no. 1. pp. 2516–2523.

[19] W. Fan et al., "Graph neural networks for social recommendation," in *Proc. World Wide Web Conf.*, 2019, pp. 417–426.

[20] L. Wu, P. Sun, Y. Fu, R. Hong, X. Wang, and M. Wang, "A neural influence diffusion model for social recommendation," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 235–244.

[21] L. Yang, Z. Liu, Y. Dou, J. Ma, and P. S. Yu, "Consisrec: Enhancing GNN for social recommendation via consistent neighbor aggregation," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 2141–2145.

[22] H. Fu et al., "Lrcbert: Latent-representation contrastive knowledge distillation for natural language understanding," in *Proc. AAAI Conf. Artif. Intell.*, 2021, vol. 35, no. 14, pp. 12 830–12 838.

[23] R. Chen, Z. Cai, and J. Yuan, "Uiesc: An underwater image enhancement framework via self-attention and contrastive learning," *IEEE Trans. Ind. Informat.*, vol. 19, no. 12, pp. 11701–11711, Dec. 2023.

[24] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. IEEE Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[25] X. Zhou, A. Sun, Y. Liu, J. Zhang, and C. Miao, "Selfcf: A simple framework for self-supervised collaborative filtering," in *Proc. ACM Trans. Recomm. Syst.*, 2023, pp. 1–25. [Online]. Available: https://arxiv.org/pdf/2107.03019.pdf

[26] Y. Yang, C. Huang, L. Xia, and C. Li, "Knowledge graph contrastive learning for recommendation," in *Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2022, pp. 1434–1443.

[27] J. Yu, H. Yin, X. Xia, T. Chen, L. Cui, and Q. V. H. Nguyen, "Are graph augmentations necessary? simple graph contrastive learning for recommendation," in *Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2022, pp. 1294–1303.

[28] D. Zou et al., "Multi-level cross-view contrastive learning for knowledge-aware recommender system," in *Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2022, pp. 1358–1368.

[29] Y. Ma, Y. He, A. Zhang, X. Wang, and T.-S. Chua, "Crosscbr: Cross-view contrastive learning for bundle recommendation," in *Proc. 28th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2022. pp. 1233–1241.

[30] F. Sun et al., "Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer," in *Proc. 28th ACM Conf. Inf. Knowl. Manage.*, 2019, pp. 1441–1450.

[31] E. Alsentzer, S. Finlayson, M. Li, and M. Zitnik, "Subgraph neural networks," *Adv. Neural Inf. Proces. Syst.*, vol. 33, pp. 8017–8029, 2020.

[32] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," 2012. [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1205/1205.2618.pdf

[33] C. Chen, M. Zhang, Y. Liu, and S. Ma, "Social attentional memory network: Modeling aspect-and friend-level differences in recommendation," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2019, pp. 177–185.

[34] Q. Wu et al., "Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems," in *Proc. World Wide Web Conf.*, 2019, pp. 2091–2102.

[35] L. Chen, L. Wu, R. Hong, K. Zhang, and M. Wang, "Revisiting graph based collaborative filtering: A linear residual graph convolutional network approach," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 01, pp. 27–34.

[36] K. Yuan, G. Liu, J. Wu, and H. Xiong, "Semantic and structural view fusion modeling for social recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 11, pp. 11872–11884, Nov. 2023.

[37] C. Wang et al., "Towards representation alignment and uniformity in collaborative filtering," in *Proc. 28th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2022, pp. 1816–1825.

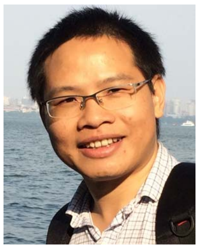
Chuyuan Wei (Member, IEEE) received the Ph.D. degree in computer science from the Beijing Institute of Technology, Beijing, China, in 2016.

He is currently an Associate Professor with the College of Electrical and Information Engineering, Beijing University of Civil Engineering and Architecture, Beijing, China. His current research interests include machine learning, data mining, and natural language processing. He has authored or coauthored some academic papers in international journals and conferences such as EMNLP, IJCIS, Sensors and Materials, and *Chinese Journal of Electronics*.

Dr. Wei is a Senior Member of the China Computer Federation.

Chuanhao Hu received the B.S. degree in software engineering from Linyi University, Linyi, China, in 2021. He is currently working toward the master's degree in mechanical engineering with the Beijing University of Civil Engineering and Architecture, Beijing, China.

His research interests include recommendation systems, specifically exploring recommendation methodologies based on graph neural networks.



Chang-Dong Wang (Senior Member, IEEE) received the Ph.D. degree in computer science from Sun Yat-sen University, Guangzhou, China, in 2013.

From January 2012 to November 2012, he was a Visiting Student with the University of Illinois at Chicago. He is currently an Associate Professor with the School of Data and Computer Science, Sun Yat-sen University. His current research interests include machine learning and data mining. He has authored or coauthored more than 120 scientific papers in international journals and conferences such as IEEE TPAMI, IEEE TKDE, IEEE TCYB, IEEE TNNLS, ACM TKDD, IEEE TSMC-Systems, IEEE TII, IEEE TSMC-C, KDD, AAAI, IJCAI, CVPR, ICDM, CIKM, and SDM.

Dr. Wang was the recipient of the 2015 Chinese Association for Artificial Intelligence (CAAI) Outstanding Dissertation. His ICDM 2010 paper won the Honorable Mention for Best Research Paper Awards. He won the 2012 Microsoft Research Fellowship Nomination Award. He is an Associate Editor for *Journal of Artificial Intelligence Research (JAIR)*.

Shuqiang Huang received the Ph.D. degree in computer science from the South China University of Technology, Guangzhou, China, in 2010.

He is currently a Full Professor with the College of Cyber Security of Jinan University, Jinan University, Guangzhou, China, and also with the Guangdong Key Laboratory of Data Security and Privacy Preserving, Guangzhou. His main research interests include edge computing, industrial IoT, and artificial intelligence. He has authored or coauthored more than 50 academic papers in international journals such as IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, *TCS*, and ACM TRANSACTIONS ON INTELLIGENT SYSTEMS AND TECHNOLOGY.

Dr. Huang is a Distinguished Member of the China Computer Federation.