Predictive and Contrastive: Dual-Auxiliary Learning for Recommendation

Yinghui Tao, Min Gao, Junliang Yu, Zongwei Wang, Qingyu Xiong, Xu Wang

***Abstract*—Self-supervised learning (SSL) recently has achieved outstanding success on recommendation. By setting up an auxiliary task (either predictive or contrastive), SSL can discover supervisory signals from the raw data without human annotation, which greatly mitigates the problem of sparse user-item interactions. However, most SSL-based recommendation models rely on general-purpose auxiliary tasks, e.g., maximizing correspondence between node representations learned from the original and perturbed interaction graphs, which are explicitly irrelevant to the recommendation task. Accordingly, the rich semantics reflected by social relationships and item categories, which lie in the recommendation data-based heterogeneous graphs, are not fully exploited. To explore recommendation-specific auxiliary tasks, we first quantitatively analyze the heterogeneous interaction data and find a strong positive correlation between the interactions and the number of user-item paths induced by meta-paths. Based on the finding, we design two auxiliary tasks that are tightly coupled with the target task (one is predictive and the other one is contrastive) towards connecting recommendation with the self-supervision signals hiding in the positive correlation. Finally, a model-agnostic DUal-Auxiliary Learning (DUAL) framework which unifies the SSL and recommendation tasks is developed. The extensive experiments conducted on three real-world datasets demonstrate that DUAL can significantly improve recommendation, reaching the state-of-the-art performance.**

***Index Terms*—Recommender system, Self-supervised learning, Heterogeneous graph.**

I. Introduction

MOST modern recommender system models are based on deep neural architectures, which require a large volume of training data to take full advantage of their capacity. However, since users can only interact with a fraction number of provided items, the observed user behavioral data is extremely sparse, making deep neural recommendation models struggle to learn high-quality representations [27]. Recently, self-supervised learning (SSL) becomes a latest trend in multiple fields. Due to its effectiveness to discover supervisory signals from the raw data without human annotation, it is inherently a sliver bullet to the data sparsity issue, and has gained attention from the recommendation community [29].

Some early research effort on SSL has transplanted the successful ideas from other fields to recommendation. For example, Zhou *et al.* [41] introduce SSL into the sequential recommendation to learn the intrinsic correlation of data through maximizing the mutual information of context information in different forms or granularities. Wu *et al.* [27] perform stochastic augmentation by perturbing the raw graph with edge dropout or random feature masking to create supplementary views and then maximize the agreement between the representations of the same node but learned from different views. The common thought of these works is to set up an auxiliary task for mining self-supervision signals. However, despite their decent performance, since these auxiliary tasks are derived from other scenarios such as language modeling and graph learning, they cannot seamlessly be generalized to the recommendation scenario. 从用户到某个物品直接的路径越多越可能点击

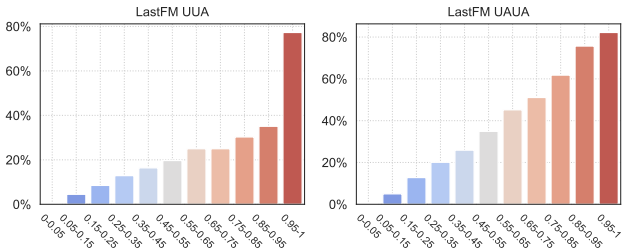


Fig. 1. The positive correlation between user-item interactions and the induced path numbers via meta-paths. The horizontal axis represents the link-score derived from the number of path instances between the head user and the tail artist; the vertical axis represents the possibilities with which the head user consumed the tail item in the paths.

The real-world graph usually comes with multi-types of nodes and edges, also widely known as heterogeneous graphs [18]. However, most research on self-supervised learning for recommender systems focuses on homogeneous graphs with a single type of node and edge. Accordingly, the rich semantics lying in heterogeneous graphs constructed from recommendation data, such as users' social relationships and item categories, are not fully exploited in these tasks. As a typical instantiation of heterogeneous graphs, the meta-path is introduced in search of higher-level semantic representations. Consequently, we capture the semantic information by deriving a variety of meta-paths starting with a head node (user), passing through several objects, and then terminating at a tail node (item). To design recommendation-specific auxiliary tasks, we first quantitatively analyze the interaction data of

This work was supported by National Key R&D Program of China (2018YFB1403602), the National Natural Science Foundation of China (62176028), and the Overseas Returnees Innovation and Entrepreneurship Support Program of Chongqing (cx2020097). (*Corresponding author: Min Gao.*)

Yinghui Tao, Min Gao, Zongwei Wang, and Qingyu Xiong are with the Key Laboratory of Dependable Service Computing in Cyber Physical Society (Chongqing University), Ministry of Education and School of Big Data and Software Engineering, Chongqing 401331, China (e-mail: taoyinghui@cqu.edu.cn; gaomin@cqu.edu.cn; zongwei@cqu.edu.cn; cquxqy@163.com).

Junliang Yu is with the School of Information Technology and Electrical Engineering, The University of Queensland, Queensland 4072, Australia (e-mail: jl.yu@uq.edu.au).

Xu Wang is with the College of Mechanical and Vehicle Engineering, Chongqing University, Chongqing 400044, China (e-mail: wx921@163.com).

the public music recommendation dataset Last-FM and find a strong positive correlation between the interactions and the numbers of user-item paths induced by meta-paths. Fig.1 shows the analysis of meta-paths User-User-Artist (UUA) and User-Artist-User-Artist (UAUA). Consistent with intuition, the higher the number of path instances between a user and an artist, the higher the likelihood of an interaction between them. Based on this finding, we innovatively design a path-regression task to predict the link-score, which was extracted from the number of meta-path instances. More precisely, we construct the commuting matrix [19] of a specific meta-path instances, normalize it to obtain the link-score. Subsequently, the link-score is used as the self-supervision signal for a path-regression task, fully integrating semantic information and improving recommendation.

We obtain two discriminative representations for each node through the path-regression and recommendation tasks. For the sake of refining the essential information existing in the original user-item interaction graph, we employ a contrastive objective that distinguishes the representations derived from the path-regression and recommendation tasks to make representations more accurate. The path-regression task belongs to the predictive task of self-supervised learning [29]. By combining two types of self-supervised tasks (predictive and contrastive tasks), our model self-generates informative supervision and handles the data-label relationships on the one hand, and deals with the inter-data information (data-data pairs) on the other hand.

To unify the recommendation task and the two auxiliary tasks, we jointly optimize their objectives. However, the competition and conflict between different tasks may lead to performance deterioration, namely negative transfer [22]. In light of this, we propose a self-supervised Dual-Auxiliary Learning framework (DUAL) in this paper. The framework adopts the Customized Gate Control [20] as the bottom architecture, which explicitly separates shared and task-specific components to avoid inherent conflicts and negative transfer, so that the recommendation performance can be significantly improved. The major contributions of this paper are summarized as follows:

- We analyze the interaction data and find a strong positive correlation between the interactions and the numbers of user-item paths induced by meta-paths, which provides reliable self-supervision signals for model training.
- We propose a model-agnostic self-supervised dual-auxiliary learning framework. It is with good extensibility, and can empirically adapt to different kinds of graph neural networks.
- Extensive experiments demonstrate the superiority of DUAL by comparing it with the state-of-the-art supervised and self-supervised models. Furthermore, this work experimentally verifies that jointly applying path-regression predictive and contrastive tasks can fully integrate semantic information and significantly improve model performance.

II. Related Work

A. Self-Supervised Learning in Recommender Systems

Most work in recommender systems has focused on supervised or semi-supervised learning settings, which require adequate user-item interaction data for model training. However, interaction data is often limited, expensive, and inaccessible. These supervised or semi-supervised methods are challenging to adapt to real-world scenarios due to their heavy reliance on the quantity and quality of interactions.

In such cases, self-supervised learning [2, 9] emerges as the times require. It aims to train a network on an auxiliary objective where the ground-truth samples are automatically obtained from the raw data, withdrawing the need for excessively user-item interactions. Based on how the auxiliary tasks are designed, Xie *et al.* [29] divide the SSL methods into two categories, namely contrastive models and predictive models. The major difference between the two categories is that contrastive models require data-data pairs for training, while predictive models require data-label pairs, where the labels are self-generated from the data. The contrastive methods deal with the inter-data information (data-data pairs), while the predictive methods aim to self-generate informative labels from the data as supervision and handle the data-label relationships.

Inspired by the success of self-supervised learning, some recent work [32, 14, 41, 31] has transplanted the same idea to the scenario of recommendation. Yao *et al.* [32] propose a two-tower DNN architecture with uniform feature masking and dropout for self-supervised item recommendation. Ma *et al.* [14] mine extra signals for supervision by looking at the longer-term future and reconstructing the future sequence for self-supervision, which is essentially adopting feature masking. Lee *et al.* [12] adopt two distinct encoder networks that learn from each other and randomly generate the augmented views of each positive interaction, then further trains the model by self-supervision. Zhang *et al.* [36] propose a double-scale node dropout strategy to create self-supervision signals that can regularize user representations with different granularities against the sparsity issue. Xia *et al.* [28] propose a novel dual channel hypergraph convolutional network for session-based recommendation, which can capture the beyond pairwise relations among items and the cross-session information through hypergraph modeling. Yu [34] *et al.* innovatively integrate self-supervised learning into the training of the hypergraph convolutional network for the social recommendation, regaining the connectivity information with hierarchical mutual information maximization. Besides, some works attempt to utilize side-information of items, e.g., item attributes, in pretraining with mutual information maximization [41] and graph neural network [31]. Unlike previous works that merely focus on general-purpose contrastive tasks, we designed two auxiliary tasks closely related to the recommendation to fully integrate rich semantic information reflected by social relationships and item categories during model training.

B. Recommender Systems Based on Meta-path

Meta-path [19] is a structure to capture the semantics and has been widely used in recommender systems. Early works

[13, 40] utilize meta-path-based similarities to regularize user/item representations so that more similar user-user/item-item pairs are enforced to be closer in the latent space. Some recent works [35, 17, 38] first learn user/item representations along each path and then fuse them. Hwang *et al.* [10] propose to assist recommendation by predicting whether user-item pairs can be connected through specific meta-paths. However, these models lack explicit feedback to guide the training of models. Implicit feedback is binary and naturally noisy. The interactions between users and an item do not mean users like the item, which may be due to a mistake click. On the contrary, users might not click on an item because they did not notice it, which does not mean they will alienate it. Compared with implicit feedback, explicit feedback reflects users' preferences for items in a more fine-grained manner.

Consistent with previous work, we likewise utilize meta-paths to support learning user preferences. Nonetheless, unlike previous works, our proposed model automatically provides valuable explicit feedback for model training.

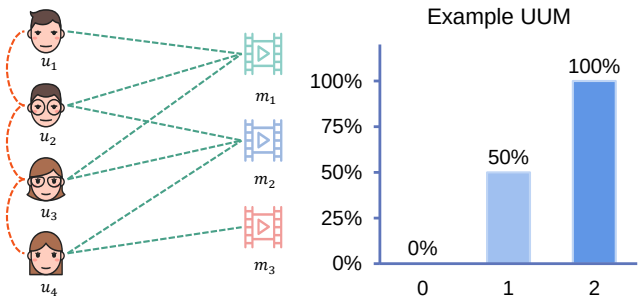
C. Multi-Task Learning in Recommender Systems

In this work, we apply the joint training of target and auxiliary tasks to optimize our model. Multi-task learning (MTL) [39] learns multiple tasks simultaneously in one single model and is proven to improve learning efficiency through information sharing between tasks [20]. MTL has been widely applied to recommender systems to exploit various user behaviors better and achieved substantial improvement. Multiple related tasks are trained together, and they can share the information they learn during the training process.

Some recommender systems have applied MTL models with efficient shared learning mechanisms. One idea is to control how Expert modules are shared across all tasks at the bottom of the multi-task model [20], and tower modules at the top handle each task separately. Ma *et al.* [15] introduce Mixture-of-Experts to MTL and propose the Multi-gate Mixture-of-Experts (MMOE) model by the gating networks assembling the experts for different tasks. Zhao *et al.* [39] explore various soft-parameter sharing techniques such as MMOE to optimize for multiple ranking objectives for video recommendation efficiently. Unlike MMOE that treats all experts equally without differentiation, Tang *et al.* [20] propose a Customized Gate Control (CGC) model to explicitly separate task-shared experts and task-specific experts. CGC is a state-of-the-art MTL method. It improves shared learning efficiency and addresses negative transfer further [22]. Accordingly, we take it as the basic framework of our model.

III. Motivation

Because heterogeneous graphs contain more comprehensive information, such as users' social relationships and item categories, it has been widely used in many data mining tasks [42, 30]. As a typical instantiation of heterogeneous graphs, meta-path [19] is widely used to capture the semantics reflected from complex information. Taking the movie data in Fig.2(a) as an example, the meta-path User-User-Movie (UUM) indicates that users' preferences for movies be affected by their social connections, while User-Movie-User-Movie (UMUM) indicates that users explore new movies through other users who have seen the same film. Depending on meta-paths, the relation between nodes in the heterogeneous graph can have different semantics.



(a) An illustrative example of user-item bipartite graph. (b) An illustrative example of the motivation.

Fig. 2. The positive correlation between user-item interactions and the induced path numbers via meta-path UUM.

The recommender system model based on meta-paths has already made many attempts [13, 40, 35, 17, 38, 3]. The work of these researchers shows that meta-paths help to learn users' preferences for items. From this conclusion, we suspect that embeddings that can accurately predict the number of path instances are also helpful for the model to judge users' preferences. Take the movie data in Fig.2 as an example:

1) u_2 *has* ***2*** *friends who watched* m_1 *and* u_2 ***watched*** m_1;
2) u_3 *has* ***2*** *friends who watched* m_2 *and* u_3 ***watched*** m_2;
3) u_1 *has* ***1*** *friends who watched* m_1 *and* u_1 ***watched*** m_1;
4) u_3 *has* ***1*** *friends who watched* m_3 *but* u_3 ***did not watch*** m_3;
5) u_2 *has* ***0*** *friends who watched* m_3 *but* u_2 ***did not watch*** m_3.

As shown in Fig.2(b), the horizontal and vertical axes represent the induced path numbers and the possibilities of interactions between the head user and the tail item, respectively. It can be found that the number of path instances between user and movie following the UUM meta-path is related to whether the user to watch the movie, specifically:

- *Users with* ***two*** *friends who watched the same movie are* ***100%*** *likely to watch the movie*;
- *Users with* ***one*** *friend who watched the same movie are* ***50%*** *likely to watch the movie*;
- *Users with* ***no*** *friends who watched the same movie are* ***0%*** *likely to watch the movie*.

We perform the same analysis on the public music recommendation dataset Last-FM based on this idea. The number of path instances following a specific meta-path of user-item can be obtained by extracting the commuting matrix for the meta-path. Then we normalized the connection number in the matrix to a link-score of [0-1] to eliminate the differences among meta-paths. Fig.1 shows a similar result as the example: the more connections between user and artist following meta-path User-User-Artist (UUA) and User-Artist-User-Artist (UAUA), the higher the possibility that the user will interact with the artist. In addition, similar phenomena can also be observed

in other datasets. The experimental results confirmed the conjecture mentioned above. Therefore, we design two self-supervised auxiliary tasks for recommendation based on this phenomenon.

IV. Preliminary

The real-world graph usually comes with multi-types of nodes and edges, also widely known as a heterogeneous graph. Learning models on heterogeneous graphs requires different considerations to effectively represent the heterogeneity of their node and edge. This section first introduces relevant concepts in this work and then presents the problem statement.

A. Relevant Concepts

Definition 1: **Heterogeneous Graph** [18]. A heterogeneous graph, denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, consists of an node set $\mathcal{V}$ and a edge set $\mathcal{E}$. A heterogeneous graph is also associated with a node type mapping function $\phi : \mathcal{V} \rightarrow \mathcal{A}$ and a edge type mapping function $\psi : \mathcal{E} \rightarrow \mathcal{T}$. $\mathcal{A}$ and $\mathcal{T}$ denote the sets of predefined node types and edge types, where $|\mathcal{A}| + |\mathcal{T}| > 2$.

Definition 2: **Meta-path** [19]. A meta-path Φ is defined as a path in the form of $A_1 \xrightarrow{T_1} A_2 \xrightarrow{T_2} \cdots \xrightarrow{T_l} A_{l+1}$ (abbreviated as $A_1 A_2 \cdots A_{l+1}$), which describes a composite relation $T = T_1 \circ T_2 \circ \cdots \circ T_l$ between nodes A_1 and A_{l+1} , where $\circ$ denotes the composition operator on relations.

Definition 3: **Commuting Matrix** [19]. Given an heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, denote the set of nodes of type A_i as $\mathcal{V}_{A_i}$, and denote the adjacency matrix between nodes of type $\mathcal{V}_{A_i}$ and nodes of type $\mathcal{V}_{A_j}$ as $\boldsymbol{W}^{ij} \in \{0,1\}^{|\mathcal{V}_{A_i}| \times |\mathcal{V}_{A_j}|}$, where entry $W^{ij}_{mn} = 1$ means there is a direct edge between node m (of type $\mathcal{V}_{A_i}$) and node n (of type $\mathcal{V}_{A_j}$), $W^{ij}_{mn} = 0$ otherwise. The commuting matrix $\boldsymbol{C}$ of meta-path $\Phi = \left(A_1 \xrightarrow{T_1} A_2 \cdots \xrightarrow{T_l} A_{l+1}\right)$ is defined as $\boldsymbol{C}^{\Phi} = \boldsymbol{W}^{12}\boldsymbol{W}^{23}\cdots\boldsymbol{W}^{l(l+1)}$.

Definition 4: **Link-score Matrix**. Given the commuting matrix $\boldsymbol{C}^{\Phi} \in \mathbb{R}^{|\mathcal{V}_{user}| \times |\mathcal{V}_{item}|}$ of meta-path Φ, the link-score matrix $\boldsymbol{M}^{\Phi} \in \mathbb{R}^{|\mathcal{V}_{user}| \times |\mathcal{V}_{item}|}$ is defined as a matrix obtained by normalizing commuting matrix $\boldsymbol{C}^{\Phi}$.

As for auxiliary tasks, we use the link-score matrix $\boldsymbol{M}$ as explicit supervisory signals of auxiliary tasks during model training. Taking the commuting matrix $\boldsymbol{C}$ and link-score matrix M in Fig.2 as an example, $C^{\Phi}_{i,j}$ represents the number of path instances between i-th user and j-th item following the meta-path Φ. Clearly, $C^{UUM}_{2,1} = 2$ represents that there are 2 path instances (i.e. $u_2 \longrightarrow u_1 \longrightarrow m_1$, $u_2 \longrightarrow u_3 \longrightarrow m_1$) between u_2 and m_1 following the meta-path UUM. Similarly, $C^{UUM}_{2,2} = 1$, $C^{UUM}_{2,3} = 0$. To eliminate the magnitude difference in the number of path instances between users, we scale $C^{UUM}_{i,j}$ to a value between 0 and 1 with Min-max normalization to obtain the link-score, i.e. $M^{UUM}_{2,1} = 1$, $M^{UUM}_{2,2} = 0.5$, and $M^{UUM}_{2,3} = 0$.

B. Problem Statement

Consider a recommender system with a user-set $\mathcal{U} = \{u_1, \ldots, u_m\}$ and an item-set $\mathcal{V} = \{v_1, \ldots, v_n\}$. A sparse interaction matrix $\boldsymbol{R} \in \mathbb{R}^{m \times n}$ is used to denote all users' implicit feedback on all items, where each element r_{uv} = 1 indicates that user u has observed or interacted with item v before, and r_{uv} = 0 otherwise. The goal of the recommendation model is to to provide users with a Top-K item recommendation list based on historical interactions.

Graph-based methods usually denote the user-item bipartite graph as $\mathcal{G}_{UI} = \{(u, v)|u \in \mathcal{U}, v \in \mathcal{V}\}$, which is constructed from user-item history interaction matrix $\boldsymbol{R}$. A nonzero r_{uv} matches an edge between user u and item v on $\mathcal{G}_{UI}$. Besides, we denote the user-item bipartite graph as $\mathcal{G}_{\Phi} = \{(u, v)|M_{u,v} > 0\}$, which is constructed from the link-score matrix $\boldsymbol{M}$ of meta-path Φ. The two embedding matrices initialized for all nodes, generated by parameterizing each ID with a random initial vector, are denoted as shared embedding $\mathbf{E}_S \in \mathbb{R}^{(m+n) \times d}$ and task-specific embedding $\mathbf{E}_T \in \mathbb{R}^{(m+n) \times d}$, respectively. d is the embedding dimension.

Given a target user, our goal is to generate a ranked list over the item-set $\mathcal{V}$ for the given user.

V. The Proposed Model

In this section, we propose a novel self-supervised dual-auxiliary learning framework DUAL. The overall architecture is shown in Fig.3, which has three critical components. 1) The embedding propagation layer aggregates meta-path-based neighbors by a GNN encoder to get the semantic-specific user/item embeddings. These embeddings will be selectively fused into final node representations for the recommendation and path-regression tasks. 2) The embedding fusion layer tells the difference of meta-paths via a gating network and gets the optimally weighted combination of the semantic-specific node embedding for the recommendation and path-regression tasks. 3) The multi-task joint training layer simultaneously optimizes the recommendation and predictive (i.e., the path-regression) task, and then we maximize the agreement between node embeddings of them by a contrastive loss.

Our model is trained on three tasks, namely recommendation task, predictive task, and contrastive task. As for recommendation task, we aim to estimate the likelihoods that users will interact with items. In addition to the recommended task, our model also combines a predictive task and a contrastive task to assist model training. The predictive task predicts the link-score, i.e., the normalized number of user-item paths induced by meta-paths, while the contrastive task maximizes the agreement between node representation for recommendation and predictive tasks. To avoid the negative transfer in multi-task joint training, we adopt CGC [20] as the bottom architecture of our model.

A. Embedding Propagation Layer

The input of the embedding propagation layer consists of shared modules (blue arrows) and task-specific modules (red arrows). Specifically, shared modules in DUAL are responsible for learning shared patterns, while task-specific modules extract patterns for specific tasks. As shown in Fig.3, shared modules (or task-specific modules) consist of multiple bipartite graphs and shared embedding $\mathbf{E}_S$ (or task-specific embedding

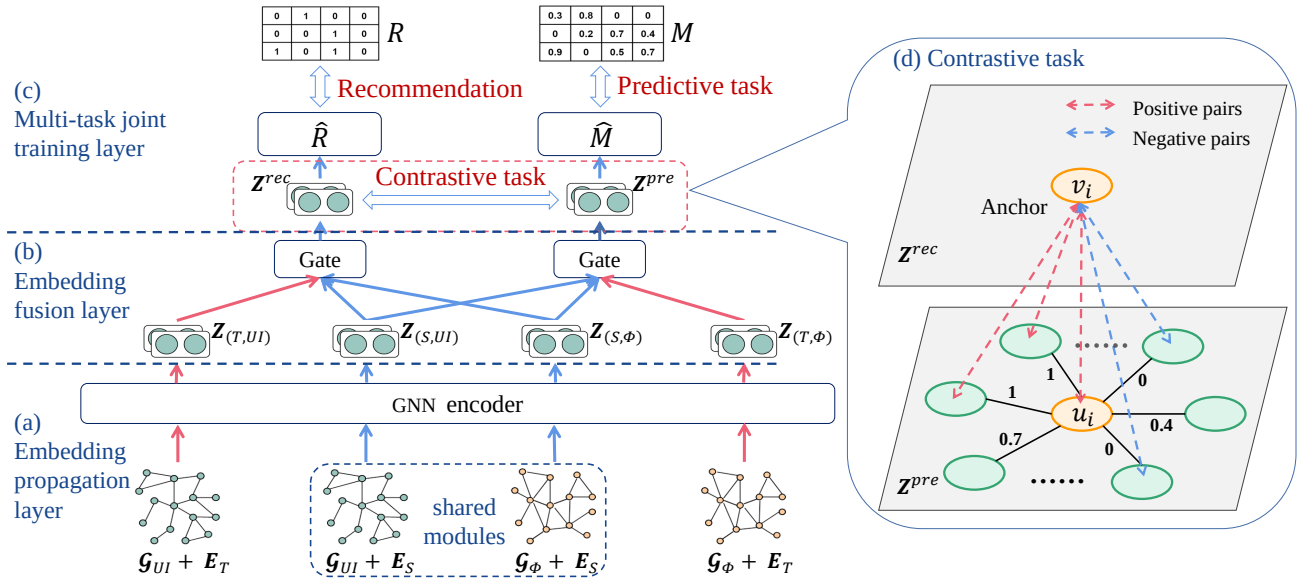


Fig. 3. The overall framework of DUAL. (a) The embedding propagation layer aims to learn the node representations by iteratively aggregating features from neighboring nodes in bipartite graphs. (b) In the embedding fusion layer, each tower absorbs knowledge from both shared modules and its own task-specific module. (c) The multi-task joint training layer aims to alculate the loss and end-to-end optimization for DUAL. (d) For the contrastive task, dada-data pairs are derived from the embeddings of predictive ($\mathbf{Z}^{pre}$) and recommendation tasks ($\mathbf{Z}^{rec}$), respectively.

$\mathbf{E}_T$). The number of bipartite graphs in modules is a hyper-parameter to tune. We use two bipartite graphs as input, one constructed through the interaction matrix ($\mathcal{G}_{UI}$) and the other constructed through a meta-path Φ starting from users and ending with items ($\mathcal{G}_\Phi$). The node embeddings after propagation of modules ($\mathbf{Z}_{(S,UI)}, \mathbf{Z}_{(S,\Phi)}, \mathbf{Z}_{(T,UI)}, \mathbf{Z}_{(T,\Phi)}$) are shown as follows:

$$(\mathbf{Z}_{(S,UI)}, \mathbf{Z}_{(S,\Phi)}) = H(\mathbf{E}_S, \mathcal{G}_{UI}, \mathcal{G}_\Phi), \tag{1}$$

$$(\mathbf{Z}_{(T,UI)}, \mathbf{Z}_{(T,\Phi)}) = H(\mathbf{E}_T, \mathcal{G}_{UI}, \mathcal{G}_\Phi), \tag{2}$$

where $H(\cdot)$ is the encoder function to encode connectivity information into representation learning. The encoder aims to learn user and item representations by iteratively aggregating features from neighboring nodes in these bipartite graphs. Since LightGCN [7] has superior performance for recommender systems, we adopt it as the basic structure of our encoder. The message passing operation (two propagation layers) with self-loops is defined as follows:

$$\mathbf{Z} = \left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}\right)\left(\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}\right)\mathbf{E}\right), \tag{3}$$

where $\hat{A} = A + I$ and $\hat{D} = D + I$. A, D, I are the adjacency matrix, the diagonal node degree matrix, and the identity matrix extracted from bipartite graph $\mathcal{G}$, respectively. I is used to integrate self-loop connections on nodes. $\mathbf{E}$ is an initialized embedding matrix ($\mathbf{E}_S$ or $\mathbf{E}_T$).

B. Embedding Fusion Layer

In the embedding fusion layer, each tower absorbs knowledge from both shared modules and its task-specific module and fuses them, which means that the parameters of shared modules are affected by both recommendation and path-regression predictive tasks. In contrast, parameters of task-specific modules are only affected by the corresponding specific task $k \in \{rec, pre\}$. rec and pre are abbreviations for recommendation and path-regression predictive tasks, respectively.

The outputs of shared modules and task k's own module are combined through a gate structure to obtain fusing weight for the specific task. The structure of the gating network is based on a single-layer feedforward network with softmax as the activation function. More precisely, the output of task k's gating network is formulated as:

$$\mathbf{Z}^k = g^k(x)S^k, \tag{4}$$

where $\mathbf{Z}^k$ is the final representations of all nodes of task k. $g^k(x)$ is a weighting function to calculate the weight vector of task k through linear transformation and a softmax layer:

$$g^k(x) = softmax\left(W^k x\right), \tag{5}$$

where $x \in \mathbb{R}^d$ is the input representation. $W^k \in \mathbb{R}^{(m_s+1)\times d}$ is a parameter matrix, and $(m_s + 1)$ is the number of shared modules and task k's own module. S^k is a selected matrix composed of all selected vectors including task k's specific module ($\mathbf{Z}_{(T,UI)}$ or $\mathbf{Z}_{(T,\Phi)}$) and shared modules ($\mathbf{Z}_{(S,UI)}$ and $\mathbf{Z}_{(S,\Phi)}$):

$$S^{rec} = [\mathbf{Z}^\top_{(T,UI)}, \mathbf{Z}^\top_{(S,UI)}, \mathbf{Z}^\top_{(S,\Phi)}]^\top, \tag{6}$$

$$S^{pre} = [\mathbf{Z}^\top_{(T,\Phi)}, \mathbf{Z}^\top_{(S,UI)}, \mathbf{Z}^\top_{(S,\Phi)}]^\top. \tag{7}$$

C. Multi-Task Joint Training Layer

The multi-task joint training layer first generates two scores indicating the user's preference for a item and the number for path instances between them. For each task k, a specific output layer is employed. We denote the representations of user u and

item v in $\mathbf{Z}^k$ as $\mathbf{p}_u^k \in \mathbb{R}^d$ and $\mathbf{q}_v^k \in \mathbb{R}^d$. The scores of user u for item v are calculated as follows:

$$\hat{R}_{uv} = \mathbf{h}^{rec\top}(\mathbf{p}_u^{rec} \odot \mathbf{q}_v^{rec}) = \sum_{i=1}^{d} h_i^{rec} p_{u,i}^{rec} q_{v,i}^{rec}, \quad (8)$$

$$\hat{M}_{uv} = \mathbf{h}^{pre\top}(\mathbf{p}_u^{pre} \odot \mathbf{q}_v^{pre}) = \sum_{i=1}^{d} h_i^{pre} p_{u,i}^{pre} q_{v,i}^{pre}, \quad (9)$$

where $\mathbf{h}^{rec} \in \mathbb{R}^d$ and $\mathbf{h}^{pre} \in \mathbb{R}^d$ denote the output layers for recommendation and the predictive task, respectively. Then for our target task – recommendation, the candidate items will be ranked in descending order of $\hat{R}_{uv}$ to provide the Top-K item recommendation list.

1) Loss Function of Recommendation: We apply the efficient non-sampling learning [1] to optimize our target task – recommendation. It is a recently proposed learning method and has been shown to be superior in both effectiveness and efficiency than traditional sampling-based learning methods. For a batch of users $\mathcal{B}$ and the whole item set $\mathcal{V}$, the recommendation task loss is

$$\mathcal{L}_{rec}(\Theta) = \sum_{u\in\mathcal{B}}\sum_{v\in\mathcal{V}^+}\left(\left(c_v^+ - c_v^-\right)\hat{R}_{uv}^2 - 2c_v^+\hat{R}_{uv}\right) + \sum_{i=1}^{d}\sum_{j=1}^{d}\left(\left(\sum_{u\in\mathcal{B}} p_{u,i}^{rec} p_{u,j}^{rec}\right)\left(\sum_{v\in\mathcal{V}} c_v^- q_{v,i}^{rec} q_{v,j}^{rec}\right)\left(h_i^{rec} h_j^{rec}\right)\right), \quad (10)$$

where c_v^+ and c_v^- are hyper-parameters that denote the weight of positive and negative instances, respectively. $\mathcal{V}^+$ is the positive samples set.

2) Loss Function of Predictive Task: As for the path-regression predictive task, we use the link-score matrix $\boldsymbol{M}$ as explicit supervisory signals. We minimize the mean squared error (MSE) between the predictions $\hat{\boldsymbol{M}}$ and the link-score matrix $\boldsymbol{M}$:

$$\mathcal{L}_{pre}(\Theta) = \sum_{u\in\mathcal{B}}\sum_{v\in\mathcal{V}}\left(M_{uv} - \hat{M}_{uv}\right)^2. \quad (11)$$

3) Loss Function of Contrastive Task: As for the path-guided contrastive task, we maximize the agreement between node representation in $\mathbf{Z}^{rec}$ and $\mathbf{Z}^{pre}$. We treat these node representations as relevant but distinct views. Then, we employ a contrastive objective that distinguishes the embeddings of positive and negative samples in the two views. Specifically, as shown in Fig.2, any node in $\mathbf{Z}^{rec}$ is treated as an anchor (v_i). In $\mathbf{Z}^{pre}$, the same anchor node (u_i) and nodes whose link-score between themselves and the anchor is 1, are positive samples. Nodes whose link-score between themselves and the anchor is 0 are naturally regarded as negative samples. Because these nodes are not closely related to the anchor, they are more probably to be truly negative instances. Our model selects positive and negative samples through the guidance of link-score, which can increase the number of positive samples and improve the credibility of negative samples.

The auxiliary supervision of positive pairs encourages the consistency between different views of the same node for prediction, while the supervision of negative pairs enforces the divergence among different nodes. Formally, we follow SGL [27] and adopt the contrastive loss, InfoNCE [6], to maximize the agreement of positive pairs and minimize that of negative pairs:

$$\mathcal{L}_{con}(\Theta) = \sum_{(i,j)\in Q^+} -\log\frac{\exp\left(s\left(\mathbf{z}_i, \mathbf{z}_j\right)/\tau\right)}{\sum_{(i,\tilde{j})\in Q^-}\exp\left(s\left(\mathbf{z}_i, \mathbf{z}_{\tilde{j}}\right)/\tau\right)}, \quad (12)$$

where Q^+ and Q^- denote the positive and negative samples set, respectively. The function $s(\cdot,\cdot)$ measures the similarity between two vectors, which is set as cosine similarity function; τ is the temperature hyperparameter.

Throughout the whole model, we are most concerned about the performance of the recommendation, and auxiliary tasks are designed to make the recommendation profit. Therefore, we can further define the loss function of the whole model as the weighted summation of the recommendation's loss and auxiliary tasks' loss as follows:

$$\mathcal{L} = \mathcal{L}_{rec} + \lambda_{pre}\mathcal{L}_{pre} + \lambda_{con}\mathcal{L}_{con}, \quad (13)$$

where λ_{pre} denotes the weight of the predictive task, and λ_{con} denotes the weight of the contrastive task. Both λ_{pre}

Algorithm 1: The overall process of DUAL

Input: User-item bipartite graphs $\mathcal{G}_{UI}$ and $\mathcal{G}_\Phi$
The weight of predictive task λ_{pre}
The weight of contrastive task λ_{con}
Output: Final node embeddings $\mathbf{Z}^{rec}$ and $\mathbf{Z}^{pre}$

Randomly initialize node embedding $\mathbf{E}_S$ and $\mathbf{E}_T$.
while *Stopping criteria is not met* **do**
  for *each epoch* **do**
    for *path = UI, Φ* **do**
      Calculate latent features $\mathbf{Z}_{(S,path)} \leftarrow H(\mathbf{E}_S, \mathcal{G}_{path})$
      Calculate latent features $\mathbf{Z}_{(T,path)} \leftarrow H(\mathbf{E}_T, \mathcal{G}_{path})$
    end
    Concatenate the learned embeddings $S^{rec} \leftarrow [\mathbf{Z}_{(T,UI)}^\top, \mathbf{Z}_{(S,UI)}^\top, \mathbf{Z}_{(S,\Phi)}^\top]^\top$
    Concatenate the learned embeddings $S^{pre} \leftarrow [\mathbf{Z}_{(T,\Phi)}^\top, \mathbf{Z}_{(S,UI)}^\top, \mathbf{Z}_{(S,\Phi)}^\top]^\top$
    for *k = rec, pre* **do**
      Calculate the weight of meta-paths $g^k \leftarrow softmax\left(W^k x\right)$
      Fuse the final embedding $\mathbf{Z}^k \leftarrow g^k(x) S^k$
    end
    Predict the probability $\hat{R}_{uv}$ according to Eq.(8)
    Predict the link-score $\hat{M}_{uv}$ according to Eq.(9)
    Evaluate $\mathcal{L}_{con}$, $\mathcal{L}_{rec}$, and $\mathcal{L}_{pre}$ according to Eq.(10) - Eq.(12)
    Jointly optimize the overall objective $\mathcal{L} \leftarrow \mathcal{L}_{rec} + \lambda_{pre}\mathcal{L}_{pre} + \lambda_{con}\mathcal{L}_{con}$
    Back propagation and update parameters
  end
end
return $\mathbf{Z}^{rec}, \mathbf{Z}^{pre}$

and λ_{con} are hyper-parameters. The overall process of DUAL is presented in Algorithm 1.

Our framework contains no parameters on any model network except for gating networks and output layers. No parameters are needed to be trained in lightGCN, and the only trainable model parameters in the whole embedding propagation layer are node embedding matrices (shared node embedding $\mathbf{E}_S$ and task-specific node embedding $\mathbf{E}_T$). Therefore, our framework retains the low computational complexity.

VI. Experiments

In this section, we demonstrate the effectiveness of DUAL by performing experiments on three real datasets compared to the state-of-the-art recommendation methods. We also perform ablation studies on the two auxiliary tasks to verify the effectiveness of the proposed hybrid model. Besides, we explore the influence of different meta-paths and encoders on experimental results. Finally, to guide the selection of parameters of our framework, we perform hyper-parameter studies on the performance of DUAL.

TABLE I
STATISTICS OF THE DATASETS.

Datasets (Density)	Relations (A-B)	Number of A	Number of B	Number of (A-B)
Last-FM (1.36%)	User-Artist	1,892	17,632	92,834
	User-User	1,892	1,892	18,802
	Artist-Artist	17,632	17,632	153,399
Yelp (0.54%)	User-Business	16,239	14,284	198,397
	User-User	16,239	16,239	158,590
	Business-Category	14,284	511	40,009
Douban-Book (0.33%)	User-Book	13,024	22,347	792,026
	User-User	12,748	12,748	169,150
	Book-Author	21,907	10,805	21,905

A. Experimental Setup

1) Datasets: We experiment with three publicly accessible datasets from different domains: Last-FM[1] dataset from music domain, Yelp[2] dataset from business domain, and Douban-book[3] dataset from book domain. The three datasets are widely used in previous studies [10, 33]. We treat a rating as an interaction record, indicating whether a user has rated an item. Following previous works [41, 37], we only keep the 5-core datasets and filter unpopular items and inactive users with fewer than five interaction records. Besides the domain variation, these three datasets also have different sparsity degrees: The Douban-book and Yelp are very sparse datasets with a density of only 0.33% and 0.54%, respectively, while the density of Last-FM is 1.36%, which is denser than that of Yelp and Douban-book. The statistics of these datasets are summarized in Table I.

[1]https://github.com/librahu/HIN-Datasets-for-Recommendation-and-Network-Embedding/tree/master/LastFM

[2]https://github.com/librahu/HIN-Datasets-for-Recommendation-and-Network-Embedding/tree/master/Yelp

[3]https://github.com/librahu/HIN-Datasets-for-Recommendation-and-Network-Embedding/tree/master/Douban Book

2) Baselines: To demonstrate the effectiveness of our DUAL model, we compare it with several state-of-the-art methods. It is worth noting that BUIR is a predictive model and SGL is a contrastive model.

- NeuMF [8]: A neural collaborative filtering method. This method uses multiple hidden layers above the element-wise and concatenation of user and item embeddings to capture their non-linear feature interactions.
- SLIMElastic [16]: A sparse linear method, which generates top-N recommendations by aggregating from user purchase/rating profiles.
- NGCF [24]: A graph-based collaborative filtering method largely follows the standard GCN [5], including the use of nonlinear activation and feature transformation. Besides, it additionally encodes the second-order feature interaction into the message during message passing.
- LINE [21]: A network embedding method, which is suitable for arbitrary types of information networks: undirected, directed, and/or weighted. The method optimizes a carefully designed objective function that preserves both the local and global network structures.
- DGCF [25]: A graph-based collaborative filtering method, which utilizes the graph disentangling module to iteratively refine the intent-aware interaction graphs and factorial representations.
- LightGCN [7]: A GCN-based general recommendation model that leverages the user-item proximity to learn node representations and generate recommendations.
- BUIR [12]: A framework for collaborative filtering to learn user and item latent representations without negative samples. BUIR employs two distinct encoder networks to address the recurring trivial constant solutions in SSL.
- SGL [27]: A framework based on SSL, which performs data augmentation on the input bipartite graph with a dropout of nodes and edges, then draw on the idea of contrastive learning to construct an SSL task.

3) Evaluation Metrics: We adopt Recall@K and NDCG@K to evaluate the performance of all methods. The two metrics have been widely used in previous recommendation studies [7, 24, 25]. Recall@K considers whether the ground truth is ranked among the top K items, while NDCG@K is a position-aware ranking metric.

4) Implementation Details: We randomly split each dataset into training (80%), validation (10%), and test (10%) sets. Moreover, we search for the optimal parameters on validation data and evaluate the model on test data. The parameters for all baseline methods are initialized as in the corresponding papers, then carefully tuned to achieve optimal performances. For a fair comparison, the embedding size is fixed to 256 for all models. Additionally, we used early stopping with patience of 20, i.e., we will stop training if the validation loss does not decrease for 20 consecutive epochs. We set the negative sampling ratio as 5 for sampling-based methods.

For the proposed DUAL, we randomly initialize parameters and optimize the model with Adam [11]. We set the learning rate to 0.001, and the dropout ratio was set to 0.3 to prevent overfitting. The weight of non-zero instances c_v^+ is set to 1 for all datasets. The negative weight c_v^- is set to 0.15, 0.2,

TABLE II
PERFORMANCE COMPARISON OF DIFFERENT METHODS ON THREE DATASETS. THE BEST PERFORMANCE AND THE SECOND BEST PERFORMANCE METHODS ARE DENOTED IN BOLD AND UNDERLINED FONTS RESPECTIVELY.

Dataset	Method	Recall@5	Recall@10	Recall@15	Recall@20	NDCG@5	NDCG@10	NDCG@15	NDCG@20
Last-FM	NeuMF	0.1782	0.2561	0.3145	0.3633	0.1804	0.2175	0.2403	0.2570
	SLIMElastic	0.1370	0.1952	0.2328	0.2660	0.1442	0.1721	0.1871	0.1986
	NGCF	0.1658	0.2456	0.3051	0.3516	0.1643	0.2021	0.2254	0.2415
	LINE	0.1703	0.2463	0.2964	0.3422	0.1709	0.2069	0.2265	0.2422
	DGCF	0.1789	0.2610	0.3252	0.3702	0.1824	0.2216	0.2461	0.2616
	LightGCN	0.1853	0.2738	0.3290	0.3745	0.1885	0.2299	0.2516	0.2675
	BUIR	0.1947	0.2735	0.3277	0.3740	0.1985	0.2352	0.2561	0.2719
	SGL	0.1845	0.2695	0.3287	0.3785	0.2128	0.2456	0.2701	0.2880
	DUAL	**0.2052**	**0.2864**	**0.3436**	**0.3930**	**0.2250**	**0.2533**	**0.2767**	**0.2941**
	Improv.	5.39%	4.60%	4.44%	3.83%	5.73%	3.14%	2.44%	2.12%
Yelp	NeuMF	0.0366	0.0582	0.0794	0.0980	0.0361	0.0423	0.0491	0.0547
	SLIMElastic	0.0253	0.0388	0.0500	0.0595	0.0248	0.0286	0.0320	0.0349
	NGCF	0.0331	0.0528	0.0732	0.0905	0.0313	0.0376	0.0439	0.0490
	LINE	0.0367	0.0545	0.0713	0.0894	0.0335	0.0388	0.0444	0.0496
	DGCF	0.0424	0.0690	0.0885	0.1089	0.0394	0.0478	0.0539	0.0598
	LightGCN	0.0430	0.0707	0.0936	0.1124	0.0426	0.0508	0.0580	0.0635
	BUIR	0.0358	0.0583	0.0782	0.0965	0.0374	0.0450	0.0520	0.0580
	SGL	0.0441	0.0700	0.0917	0.1103	0.0456	0.0537	0.0609	0.0671
	DUAL	**0.0461**	**0.0776**	**0.1033**	**0.1258**	**0.0480**	**0.0574**	**0.0656**	**0.0724**
	Improv.	4.44%	9.73%	10.32%	11.90%	5.26%	6.85%	7.68%	7.93 %
Douban-book	NeuMF	0.0615	0.0929	0.1175	0.1404	0.0764	0.0826	0.0893	0.0962
	SLIMElastic	0.0787	0.1123	0.1326	0.1494	0.1141	0.1141	0.1174	0.1215
	NGCF	0.0709	0.1053	0.1299	0.1495	0.0850	0.0920	0.0990	0.1051
	LINE	0.0753	0.1050	0.1235	0.1389	0.1009	0.1046	0.109	0.1136
	DGCF	0.0842	0.1230	0.1506	0.1734	0.1089	0.1149	0.1218	0.1287
	LightGCN	0.0917	0.1344	0.1624	0.1870	0.1197	0.1263	0.1332	0.1404
	BUIR	0.0622	0.0963	0.1235	0.1445	0.0853	0.0899	0.0968	0.1029
	SGL	0.0952	0.1372	0.1696	0.1960	0.1414	0.1445	0.1520	0.1599
	DUAL	**0.1072**	**0.1520**	**0.1827**	**0.2078**	**0.1641**	**0.1631**	**0.1683**	**0.1746**
	Improv.	12.60%	10.78%	7.72%	6.02%	16.05%	12.84%	10.72%	9.19 %

and 0.25 for Last-FM, Yelp, and Douban-book, respectively. We set the weight of the path-regression task λ_{pre} as 0.3 for Last-FM, 0.03 for Yelp and Douban-book. Due to the different loss functions of path-regression and path-guided contrastive tasks, the weight of the path-guided contrastive task λ_{con} is much greater than that of the path-regression task. Therefore, we give a lower weight to the contrastive task to balance the training rate between different tasks. Specifically, the weight of the contrastive task λ_{con} is set to 5e-4 for Last-FM, 1e-7 for Yelp, and 1e-5 for Douban-book. For Last-FM, Yelp, and Douban-book datasets, we select the optimal meta-path in [UUA, UAA, UAUA], [UUB, UBCB], and [UBAB, UBUB], respectively.

B. Performance Comparison

The performance comparison results are presented in Table II. In the experiments, we set K of Racall@K and NDCG@K as 5, 10, 15, and 20 to evaluate the DUAL and baselines. From the results, the following observations can be made:

First and foremost, the proposed DUAL achieves the best performance on the three datasets, significantly outperforming all the state-of-the-art baseline methods. The average improvement of our model to the best baseline is 3.96% on Last-FM dataset, 8.02% on Yelp dataset, and 10.74% on Douban-book dataset, which verifies the effectiveness of our model. The improvements on Yelp and Douban-book are more significant than those on the Last-FM dataset, which we hold due to the datasets' different densities. Yelp and Douban-book datasets are more sparse and lack of interaction data for model training compared with the Last-FM dataset.

The substantial improvements can be attributed to three reasons: 1) We adopt a multi-task sharing structure, which can explicitly separate shared components and task-specific components, extract and separate semantic knowledge reflected by social relationships and item categories, and improve the efficiency of joint representation learning across tasks in a general setup. 2) The proposed model can accurately capture the strong correlation between user-item interactions and the number of path instances following the specific meta-path. 3) Combining different types of SSL tasks can provide more semantic supervisory signals for recommendations, which is essential in improving the model's performance.

Second, methods based on LightGCN generally outperform other graph-based. For example, in Table II, the performance of LightGCN, SGL, and DUAL are better than NGCF and DGCF. This is consistent with previous work [7], which indicates the strong performance of LightGCN in the recommendation. LightGCN has such powerful performance because it is the state-of-the-art graph-based CF method that devises a light graph convolution to ease the training difficulty and pursue better generation ability.

Third, self-supervised learning methods (BUIR, SGL, and DUAL) generally perform better than supervised methods. In addition, it can be seen that our hybrid approach that jointly applies contrastive task and predictive task significantly outperforms the two models that only use one standalone predictive task (BUIR) or contrastive task (SGL). It verifies

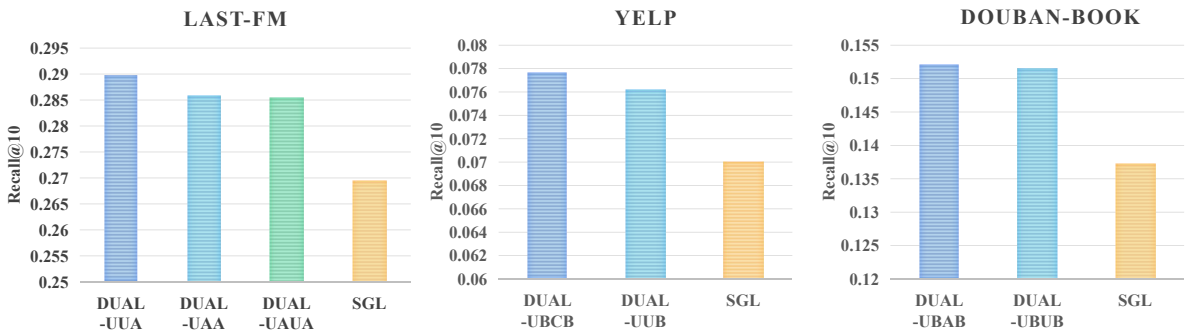


Fig. 4. Impacts of different meta-paths. We abbreviate user and artist for the Last-FM to U and A, respectively. For the Yelp, we abbreviate user, businesses, and category to U, B, and C, respectively. For the Douban-book, we abbreviate user, book, and author to U, B, and A, respectively.

the effectiveness of both types of auxiliary tasks in modeling user preferences. Regarding the poor performance of the self-supervised model BUIR on Yelp and Douban-book datasets, we speculate that because BUIR is only trained with positive samples, it does not work well on extremely sparse datasets.

C. Ablation Study

In this section, we perform ablation studies on the path-regression task and path-guided contrastive task to verify the effectiveness of the proposed DUAL model. Since the contrastive task is based on the predictive task, the contrastive task is also removed in the variant that eliminates the predictive task.

- DUAL-PC: The variant model of DUAL without predictive and contrastive tasks.
- DUAL-C: The variant model of DUAL without contrastive task.

TABLE III
PERFORMANCE OF VARIANTS OF DUAL ON LAST-FM DATASET.

Model	recall@10	recall@20	ndcg@10	ndcg@20
DUAL-PC	0.2823	0.3875	0.2477	0.2896
DUAL-C	0.2845	0.3906	0.2498	0.2910
DUAL	**0.2864**	**0.3930**	**0.2533**	**0.2941**

Table III shows the performance of different variants. As shown in the table, the predictive task or the contrastive task leads to better recommendation performance. When using all the two SSL tasks, the performance of our DUAL is further improved. It is verified that the combination of path-regression and edeg-guided contrastive task is more conducive to recommendation than a single type of SSL task.

Negative transfer is a common phenomenon in multi-task learning due to the complex and competing task correlation in real-world recommender systems, especially for loosely correlated tasks [22]. However, our proposed model DUAL has a multi-task sharing structure that explicitly separates shared and task-specific components to avoid the negative transfer phenomenon. The experimental results verify that our proposed model DUAL can learn multiple tasks simultaneously in one single model and improve learning efficiency through information sharing between tasks.

D. Impacts of Different Meta-Paths

To further evaluate our DUAL model, we conduct an experiment on impacts of using different meta-paths. The result is illustrated in Fig.4, where we can observe that DUAL performs better with the addition of different meta-paths than the best performing baseline SGL, which means that incorporating meta-paths into the model is effective for improving recommendations. The proposed model DUAL can accurately capture the strong correlation between user-item interactions and the number of connections following the specific meta-paths. Furthermore, different meta-paths have tiny different performance improvements. For example, for the Last-FM dataset, users have similar preferences with their friends connected by social relationships (UUA). For the Yelp dataset, users are more interested in businesses within the same category (UBCB), while for the Douban-book dataset, users are more inclined to select books by the same author as the books they have previously read (UBAB).

E. Impacts of Different Encoders

Architecturally, the proposed DUAL is model-agnostic so as to boost a multitude of graph neural recommendation models. We evaluate our methods with five graph neural networks: GCN [5], GAT [23], TAGCN [4], SGC [26], and LightGCN [7]. As we expected, the results in Table IV show that the efficacy of our proposed methods differs depending on graph encoders. However, it is worth noting that three variants of our model (DUAL-GCN, DUAL-SCG, and DUAL) have better performance than the best baseline SGL. It shows that our proposed model DUAL has excellent extensibility and versatility and can be easily transplanted to other graph neural networks to achieve excellent recommendation performance. In particular, the model effect based on LightGCN is outstanding. That is because LightGCN, as a graph neural network specially designed for the recommender system, removes the useless or harmful operations and is more suitable for the field of recommendation.

F. Hyper-Parameter Sensitivity

To guide the selection of parameters of our framework, we perform hyper-parameter studies on the performance of DUAL. In the implementation, we use the Last-FM as the

TABLE IV
PERFORMANCE OF DUAL TRAINED BY VARIOUS ENCODERS ON THE LAST-FM DATASET.

Variants	recall@10	recall@20	ndcg@10	ndcg@20
DUAL-TAGCN	0.2523	0.3487	0.2178	0.2556
DUAL-GAT	0.2520	0.3522	0.2190	0.2579
DUAL-GCN	0.2831	0.3861	0.2485	0.2879
DUAL-SGC	0.2806	0.3878	0.2489	0.2899
DUAL (LightGCN)	**0.2864**	**0.3930**	**0.2533**	**0.2941**
SGL	0.2695	0.3787	0.2456	0.2880

evaluated dataset and LightGCN as the backbone of DUAL. We investigate the performance changes of our framework with regard to hyper-parameters on the number of layers and the link-score of negative samples for contrastive tasks.

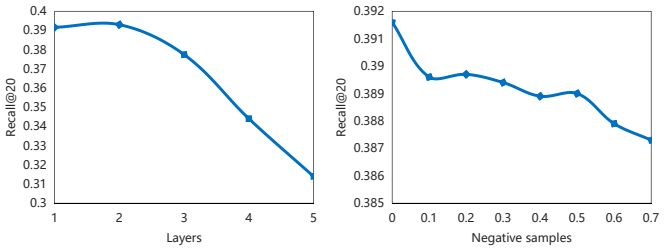


Fig. 5. Performance of DUAL with regard to the number of layers and the link-score of negative samples on the Last-FM dataset.

For the number of layers in the graph-based models, LightGCN, we select its range in the range of [1, 2, 3, 4, 5] and plot the results in Fig.5. The result shows the performance of our framework is relatively more sensitive to the number of layers than the link-score of negative samples. From Fig.5, we can see that by increasing the depth of DUAL from one to two, the recommendation results are improved. Generally, two propagation layers are sufficient to capture the heterogeneous signals. DUAL suffers the over-smoothed problem when the layer number exceeds two.

For the link-score of negative samples for the contrastive task, we select it in the range of [0, 0.1,...,0.7]. As can be observed from Fig.5, DUAL is sensitive to the link-score of negative samples, and we need to choose it carefully for the best performance. Generally, a small value of the link-score can lead to a desirable performance. This is because a large link-score between a user and an item means that they are more closely related. Therefore, the probability that the item is a potential positive is higher. Selecting these samples as negative samples will introduce noise into the model training, which degrades the recommendation performance. Our model provides a new scheme for selecting negative samples, i.e., by adjusting the link-score of negative samples to avoid selecting potentially positive samples. The experimental results also verify that the method is effective.

VII. CONCLUSION AND FUTURE WORK

In this work, we propose a self-supervised dual-auxiliary learning framework (DUAL), which automatically provides the additional supervisory signals and fully integrates rich semantic information reflected by social relationships and item categories to improve the model's performance. In our approach, we extract the commuting matrix of a specific meta-path and normalize it to obtain the link-score. Through the link-score, the path-regression predictive and path-guided contrastive tasks can be naturally closely integrated. Additionally, our proposed model DUAL effectively combines two auxiliary tasks while avoiding negative transfer. Experimental results on three public datasets show DUAL outperforms existing state-of-the-art supervised and SSL recommendation methods.

In the future, we will investigate how to design a simplified self-supervised learning framework with multiple meta-paths. We will also consider applying our approach to more complex recommendation tasks, such as sequential or conversational recommendations.

ACKNOWLEDGMENT

This work was supported by National Key R&D Program of China (2018YFB1403602), the National Natural Science Foundation of China (62176028), and the Overseas Returnees Innovation and Entrepreneurship Support Program of Chongqing (cx2020097).

REFERENCES

[1] C. Chen, M. Zhang, Y. Zhang, Y. Liu, and S. Ma. "Efficient neural matrix factorization without sampling for recommendation". In: *ACM Trans. Inf. Syst.* 38.2 (2020), 14:1–14:28.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *NAACL-HLT*. Association for Computational Linguistics, 2019, pp. 4171–4186.

[3] S. Dhelim, H. Ning, N. Aung, R. Huang, and J. Ma. "Personality-aware product recommendation system based on user interests mining and metapath discovery". In: *IEEE Trans. Comput. Soc. Syst.* 8.1 (2021), pp. 86–98.

[4] J. Du, S. Zhang, G. Wu, J. M. Moura, and S. Kar. "Topology adaptive graph convolutional networks". In: *CoRR* (2017). arXiv: 1710.10370.

[5] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. "Neural message passing for quantum chemistry". In: *ICML*. Vol. 70. PMLR, 2017, pp. 1263–1272.

[6] M. Gutmann and A. Hyvärinen. "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models". In: *AISTATS*. Vol. 9. JMLR.org, 2010, pp. 297–304.

[7] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang. "Lightgcn: Simplifying and powering graph convolution network for recommendation". In: *SIGIR*. ACM, 2020, pp. 639–648.

[8] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. "Neural collaborative filtering". In: *WWW*. ACM, 2017, pp. 173–182.

[9] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio. "Learning deep representations by mutual information estimation and maximization". In: *ICLR*. OpenReview.net, 2019.

[10] D. Hwang, J. Park, S. Kwon, K.-M. Kim, J.-W. Ha, and H. J. Kim. "Self-supervised auxiliary learning with meta-paths for heterogeneous graphs". In: *NeurIPS*. 2020.

[11] D. P. Kingma and J. Ba. "Adam: A method for stochastic optimization". In: *ICLR (Poster)*. 2015.

[12] D. Lee, S. Kang, H. Ju, C. Park, and H. Yu. "Bootstrapping user and item representations for one-class collaborative filtering". In: (2021), pp. 1513–1522.

[13] C. Luo, W. Pang, Z. Wang, and C. Lin. "Hete-cf: Social-based collaborative filtering recommendation using heterogeneous relations". In: *ICDM*. IEEE Computer Society, 2014, pp. 917–922.

[14] J. Ma, C. Zhou, H. Yang, P. Cui, X. Wang, and W. Zhu. "Disentangled self-supervision in sequential recommenders". In: *SIGKDD*. ACM, 2020, pp. 483–491.

[15] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, and E. H. Chi. "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts". In: *SIGKDD*. ACM, 2018, pp. 1930–1939.

[16] X. Ning and G. Karypis. "Slim: Sparse linear methods for top-n recommender systems". In: *ICDM*. IEEE Computer Society, 2011, pp. 497–506.

[17] C. Shi, B. Hu, W. X. Zhao, and S. Y. Philip. "Heterogeneous information network embedding for recommendation". In: *IEEE Trans. Knowl. Data Eng.* 31.2 (2019), pp. 357–370.

[18] Y. Sun and J. Han. "Mining heterogeneous information networks: A structural analysis approach". In: *SIGKDD Explor.* 14.2 (2012), pp. 20–28.

[19] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu. "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks". In: *Proc. VLDB Endow.* 4.11 (2011), pp. 992–1003.

[20] H. Tang, J. Liu, M. Zhao, and X. Gong. "Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations". In: *RecSys*. ACM, 2020, pp. 269–278.

[21] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. "Line: Large-scale information network embedding". In: *WWW*. ACM, 2015, pp. 1067–1077.

[22] L. Torrey and J. Shavlik. "Transfer learning". In: *Handbook of research on machine learning applications and trends: Algorithms, methods, and techniques*. IGI global, 2010, pp. 242–264.

[23] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. "Graph attention networks". In: *ICLR (Poster)*. OpenReview.net, 2018.

[24] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua. "Neural graph collaborative filtering". In: *SIGIR*. ACM, 2019, pp. 165–174.

[25] X. Wang, H. Jin, A. Zhang, X. He, T. Xu, and T.-S. Chua. "Disentangled graph collaborative filtering". In: *SIGIR*. ACM, 2020, pp. 1001–1010.

[26] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger. "Simplifying graph convolutional networks". In: *ICML*. Vol. 97. PMLR, 2019, pp. 6861–6871.

[27] J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, and X. Xie. "Self-supervised graph learning for recommendation". In: *SIGIR*. ACM, 2021, pp. 726–735.

[28] X. Xia, H. Yin, J. Yu, Q. Wang, L. Cui, and X. Zhang. "Self-supervised hypergraph convolutional networks for session-based recommendation". In: *AAAI*. AAAI Press, 2021, pp. 4503–4511.

[29] Y. Xie, Z. Xu, J. Zhang, Z. Wang, and S. Ji. "Self-supervised learning of graph neural networks: A unified review". In: *CoRR* (2021). arXiv: 2102.10757.

[30] C. C. Yang and L. Jiang. "Enriching user experience in online health communities through thread recommendations and heterogeneous information network mining". In: *IEEE Trans. Comput. Soc. Syst.* 5.4 (2018), pp. 1049–1060.

[31] S. Yang, Y. Liu, C. Lei, G. Wang, H. Tang, J. Zhang, and C. Miao. "A Pre-training Strategy for Recommendation". In: *CoRR* (2020). arXiv: 2010.12284.

[32] T. Yao, X. Yi, D. Zhiyuan Cheng, F. Yu, T. Chen, A. Menon, L. Hong, E. H. Chi, S. Tjoa, J. Kang, et al. "Self-supervised learning for deep models in recommendations". In: *CoRR* (2020). arXiv: 2007.12865.

[33] J. Yu, H. Yin, M. Gao, X. Xia, X. Zhang, and N. Q. V. Hung. "Socially-aware self-supervised tri-training for recommendation". In: *SIGKDD*. ACM, 2021, pp. 2084–2092.

[34] J. Yu, H. Yin, J. Li, Q. Wang, N. Q. V. Hung, and X. Zhang. "Self-supervised multi-channel hypergraph convolutional network for social recommendation". In: *WWW*. ACM, 2021, pp. 413–424.

[35] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han. "Personalized entity recommendation: A heterogeneous information network approach". In: *WSDM*. ACM, 2014, pp. 283–292.

[36] J. Zhang, M. Gao, J. Yu, L. Guo, J. Li, and H. Yin. "Double-scale self-supervised hypergraph learning for group recommendation". In: *CIKM*. ACM, 2021, pp. 2557–2567.

[37] T. Zhang, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, D. Wang, G. Liu, and X. Zhou. "Feature-level deeper self-attention network for sequential recommendation". In: *IJCAI*. ijcai.org, 2019, pp. 4320–4326.

[38] H. Zhao, Q. Yao, J. Li, Y. Song, and D. L. Lee. "Meta-graph based recommendation fusion over heterogeneous information networks". In: *SIGKDD*. ACM, 2017, pp. 635–644.

[39] Z. Zhao, L. Hong, L. Wei, J. Chen, A. Nath, S. Andrews, A. Kumthekar, M. Sathiamoorthy, X. Yi, and E. Chi. "Recommending what video to watch next: A multitask ranking system". In: *RecSys*. ACM, 2019, pp. 43–51.

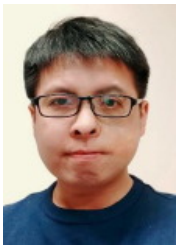
[40] J. Zheng, J. Liu, C. Shi, F. Zhuang, J. Li, and B. Wu. "Recommendation in heterogeneous information network via dual similarity regularization". In: *Int. J. Data Sci. Anal.* 3.1 (2017), pp. 35–48.

[41] K. Zhou, H. Wang, W. X. Zhao, Y. Zhu, S. Wang, F. Zhang, Z. Wang, and J.-R. Wen. "S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization". In: *CIKM*. ACM, 2020, pp. 1893–1902.

[42] X. Zhou, W. Liang, I. Kevin, K. Wang, and L. T. Yang. "Deep correlation mining based on hierarchical hybrid networks for heterogeneous big data recommendations". In: *IEEE Trans. Comput. Soc. Syst.* 8.1 (2021), pp. 171–178.

Yinghui Tao received the B.S. degree from Guizhou University, Guizhou, China, in 2020, where he is currently pursuing the master's degree with the School of Big Data & Software Engineering, Chongqing University. His research interests include recommender systems and anomaly detection.

Min Gao received the MS and PhD degrees in computer science from Chongqing University in 2005 and 2010 respectively. She is an associate professor at the School of Big Data & Software Engineering, Chongqing University. She was a visiting researcher at University of Reading and Arizona State University. Her research interests include recommendation systems, service computing, and data mining. She has published over 40 refereed journal and conference papers in these areas. She has grants from the National Natural Science Foundation of China, the China Postdoctoral Science Foundation, and the China Fundamental Research Funds for the Central Universities. She is member of the IEEE and CCF.



Junliang Yu received the B.S. and MS degree in Software Engineering from Chongqing University, Chongqing, China. Currently, he is a Ph.D. student with the school of Information Technology and Electrical Engineering at the University of Queensland, Queensland, Australia. His research interests include recommender systems and anomaly detection.

Zongwei Wang received the B.S. and MS degree in Software Engineering from Chongqing University in 2018 and 2021 respectively. His research interests include recommender systems and anomaly detection.

Qingyu Xiong received the B.S. and M.S. degree from the School of Automation, Chongqing University, in 1986 and 1991, respectively, and the Ph.D. degree from the Kyushu University of Japan in 2002. He is currently the Dean of the School of Big Data & Software Engineering, Chongqing University. His research interests include neural networks and their applications. He has authored over 100 journal and conference papers in these areas. He has over 20 research and applied grants.

Xu Wang received the M.S. and Ph.D. degrees in Mechanical Manufacturing from Chongqing University, Chongqing, China. She used to be the deputy secretary of the party committee of Chongqing University and is currently a professor at the College of Mechanical and Vehicle Engineering, Chongqing University. Her research interests include modern logistics and supply chain management, industrial Internet, service science and engineering.