Predictive and Contrastive: Dual-Auxiliary Learning for Recommendation

Yinghui Tao, Min Gao, Member, IEEE, Junliang Yu, Zongwei Wang, Qingyu Xiong, and 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) toward 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—Heterogeneous graph, recommender system, self-supervised learning (SSL).

I. INTRODUCTION

M ost 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 are extremely sparse, making deep neural recommendation models struggle to learn high-quality representations [26]. Recently, self-supervised learning (SSL) becomes the latest trend in multiple fields. Due to its effectiveness to discover supervisory signals from the raw data without human annotation, it is inherently a silver bullet to the data sparsity issue and has gained attention from the recommendation community [28].

Some early research effort on SSL has transplanted successful ideas from other fields to recommendation. For example, Zhou et al. [36] introduced SSL into the sequential recommendation to learn the intrinsic correlation of data through maximizing the mutual information between different types of data. Wu et al. [26] improved the robustness and effect of the recommendation model by generating different views for nodes and then self-discriminating them. 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.

Most real-world scenarios contain multiple node types that need to be converted into heterogeneous graphs [17]. However, much research on SSL 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. Meta-paths are introduced as a representative instantiation of heterogeneous graphs to learn multihop semantic embeddings. Consequently, we capture the semantic information by deriving various 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 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–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 [18] of specific...
meta-path instances and 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 indispensable information existing in the user–item bipartite 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 SSL [28]. 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 [21]. In light of this, we propose a self-supervised dual-auxiliary learning (DUAL) framework in this article. The framework adopts the customized gate control [19] 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 article are summarized as follows.

1) 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.
2) We propose a model-agnostic self-supervised DUAL framework. It is with good extensibility and can empirically adapt to different kinds of graph neural networks.
3) 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 the model performance.

II. RELATED WORK

A. SSL in Recommender Systems

Most work in recommender systems has focused on supervised or semisupervised learning settings, which require adequate user–item interaction data for model training. However, interaction data are often limited, expensive, and inaccessible. These works are challenging to adapt to real-world scenarios due to their severe dependence on quantity interactions.

In such cases, SSL [30], [33] emerges as the times require. It aims to train a model on an auxiliary task where signals are automatically obtained from available data, withdrawing the need for excessively user–item interactions. Based on how the auxiliary tasks are designed, Xie et al. [28] divided 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. Contrastive tasks aim to construct data–data pairs to refine the information between the data. Predictive tasks seek labels beneficial to learning node preferences from the original data and then learn the relationship between data and labels.

Inspired by the success of SSL, some recent work [13], [30], [36] has applied it to recommender systems. Yao et al. [30] added a regularization term to a large-scale item recommendation model by perturbing item embeddings. Ma et al. [13] supplemented future sequences from the perspective of long-term item sequence prediction to solve the problem of sequence recommendation. Lee et al. [10] used only positive sample pairs as input to the model, making the embeddings of users and items with interactive behaviors closer to each other. In addition, the training data are supplemented utilizing random data augmentation. Zhang et al. [33] thoroughly considered the complex relationship between users and groups, using two random data enhancements with different granularities for group recommendation scenarios. Xia et al. [27] proposed a self-supervised hypergraph convolution method for the session-based recommendation, which maximizes the mutual information between the line graph and the transformed hypergraph. Yu et al. [32] used hypergraphs to model social relations and purified user embeddings by maximizing mutual information at the user level. Besides, some works attempt to utilize side information of items, e.g., item attributes, in pretraining [36]. 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 [18] is a structure to capture the semantics and has been widely applied in recommender systems. Zheng et al. [35] utilized meta-path-based similarity to normalize node embeddings and force more similar node pairs closer. Shi et al. [16] first learned node embeddings along each path and then fuse them. Hwang et al. [8] proposed to assist recommendation by predicting whether user–item interactions are strongly correlated with the numbers 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.
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 that 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. Recommender Systems Based on Multitask Learning

In this work, we apply the joint training of target and auxiliary tasks to optimize our model. Multitask learning (MTL) [34] aims to guide model training by designing different objectives, i.e., training multiple tasks that affect each other simultaneously in a unified model [19]. MTL is widely used in recommendation scenarios to exploit various user behaviors better and achieve substantial improvement. Multiple related tasks are trained together, and they can share the information they learn during the training process.

Some studies [11], [12] jointly model the internal relationship between social behavior and purchasing behavior to promote the two behavior prediction tasks simultaneously. In addition, some recommender systems have applied MTL models with efficient shared learning mechanisms. One way is to divide the bottom layer of the model into multiple independent modules [19]. Then, the top-level module responsible for predicting specific tasks fuses the output of the bottom-level modules. Ma et al. [14] introduced mixture of experts to MTL and the multi-gate mixture-of-expert (MMoE) model to selectively fuse the outputs of the bottom-level modules by adding a gating mechanism for each task. Zhao et al. [34] analyzed the effectiveness of various soft parameter sharing methods for optimizing MTL. Unlike MMoE that treats all bottom-level modules equally and draws information from them for each task, Tang et al. [19] proposed the CGC model that divides the bottom-level modules. Some modules are public, and others only affect a specific task. CGC is a state-of-the-art MTL method. It improves shared learning efficiency and addresses negative transfer further [21]. 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, they are widely applied in various data mining scenarios [11], [12], [29], [37]. As a typical instantiation of heterogeneous graphs, meta-path [18] is widely applied 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 are 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.

Different semantic information in the recommended scenario can be extracted through the difference of node types passed by the meta-path.

The recommender system model based on meta-paths has already made many attempts [3], [16], [35]. The work of these researchers shows that meta-paths help to learn users’ preferences for items. From this conclusion, we suspect that embidgments that can accurately predict the number of path instances are also helpful for the model to judge users’ preferences.

We perform an 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 the result: the more connections between user and artist following meta-path UUA and UAU, 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.

To facilitate understanding, we take the movie data constructed artificially 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 friend who watched $m_1$ and $u_1$ watched $m_1$.
4) $u_3$ has 1 friend 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, the following conditions hold.

1) Users with two friends who watched the same movie are 100% likely to watch the movie.
2) Users with one friend who watched the same movie are 50% likely to watch the movie.

Fig. 2. Positive correlation between user–item interactions and the induced path numbers via meta-path UUM. (a) Illustrative example of user–item bipartite graph. (b) Illustrative example of the motivation.
3) Users with no friends who watched the same movie are 0% likely to watch the movie.

We design two self-supervised auxiliary tasks for recommendation based on this phenomenon.

IV. PRELIMINARY

The data extracted in real world are usually not of a single type but contain many different types of nodes. The relationships between the nodes are also diverse. It is necessary to convert these data into heterogeneous graphs and use tools such as meta-paths to model complex relationships between nodes. This section first introduces relevant concepts in this work and then presents the problem statement.

A. Relevant Concepts

Definition 1 (Heterogeneous Graph [17]): Let \( A \) be the set of node types and \( T \) be the set of edge types. The node and edge sets are denoted as \( V \) and \( E \), respectively. Then, the heterogeneous graph containing \( \phi : V \rightarrow A \) and \( \psi : E \rightarrow T \) mapping functions is denoted as \( G = (V, E) \), where \(|A| + |T| > 2\).

Definition 2 (Meta-Path [18]): A meta-path \( \Phi \) is defined on the heterogeneous graph \( G = (V, E) \), denoted as a path pattern in the form of \( A_1A_2, \ldots, A_{t+1} \), which describes the composite semantic between head node \( A_1 \) and tail node \( A_{t+1} \).

Definition 3 (Commuting Matrix [18]): A commuting matrix \( C \) is defined on the heterogeneous graph \( G = (V, E) \) and the meta-path \( \Phi = A_1A_2, \ldots, A_{t+1} \). The adjacency matrix between the node of type \( A_i \) and the node of type \( A_j \) is denoted as \( W^{ij} \in [0, 1]^{|V_A_i| \times |V_A_j|} \), where \( V_A_k \) denote the set of nodes of type \( A_k \). \( W^{mj} = 1 \) represents that there is a connection between node \( m \) of node type \( A_i \) and node \( n \) of node type \( A_j \); otherwise, \( W^{mj} = 0 \). The commuting matrix \( C \) of meta-path \( \Phi = A_1A_2, \ldots, A_{t+1} \) is defined as \( C^\Phi = W^{j2}W^{23} \ldots, W^{t+11} \).

Definition 4 (Link-Score Matrix): A link-score matrix \( M \) is defined on the heterogeneous graph \( G = (V, E) \) and the meta-path \( \Phi = A_1A_2, \ldots, A_{t+1} \). Given the commuting matrix \( C^\Phi \in \mathbb{R}^{V_A_m \times V_A_m} \) of meta-path \( \Phi \), the link-score matrix \( M^{\Phi} \in \mathbb{R}^{V_A_m \times V_A_m} \) is defined as a matrix obtained by normalizing commuting matrix \( C^\Phi \).

The paths between two nodes are infinite, but the paths that follow a particular meta-path are finite. Therefore, for arbitrary user–item pairs, the number of path instances induced by meta-path \( \Phi \) can be obtained by computing the commuting matrix \( C^\Phi \).

The data in recommender systems are often extremely sparse. Therefore, we can use sparse matrices to store and operate when we calculate the commuting matrix, i.e., only nonzero elements in the matrix are considered. In addition, the link-score matrix needs to be calculated only once and does not need to be updated iteratively. Therefore, the time consumption of calculating the link score is acceptable.

As for the auxiliary task, we use the link-score matrix \( M \) as explicit supervisory signals of auxiliary tasks during model training. Taking the commuting matrix \( C \) and link-score matrix \( M \) in Fig. 2 as an example, \( C^\Phi_{ij} \) represents the number of path instances between the \( i \)th user and the \( j \)th item following the meta-path \( \Phi \). Clearly, \( C^\Phi_{2,1} = 2 \) represents that there are two path instances (i.e., \( u_2 \rightarrow u_1 \rightarrow m_1 \) and \( u_2 \rightarrow u_3 \rightarrow m_1 \)) between \( u_2 \) and \( m_1 \) following the meta-path UUM. Similarly, \( C^\Phi_{2,2} = 1 \) and \( C^\Phi_{2,3} = 0 \). To eliminate the magnitude difference in the number of path instances between users, we normalize the commuting matrix according to rows (users) to obtain the link score, i.e., \( M_{UUM} = 1 \), \( M_{2,2} = 0.5 \), and \( M_{2,3} = 0 \). As a result, although active users have numerous meta-paths, it does not lower the link score of inactive users.

B. Problem Statement

Consider a recommender system with a user set \( \mathcal{U} = \{u_1, \ldots, u_m\} \) and an item set \( \mathcal{I} = \{i_1, \ldots, i_n\} \). A sparse interaction matrix \( R \in \mathbb{R}^{m \times n} \) 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 provide users with a Top-K item recommendation list based on historical interactions.

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

Given a target user, our goal is to generate a ranked list over the item set \( \mathcal{I} \) for the given user.

V. PROPOSED MODEL

In this section, we propose DUAL, a novel self-supervised DUAL method. The framework is shown in Fig. 3, which has three critical parts.

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 through a gating network and obtains the optimally weighted combination of the semantic-specific node embedding for the recommendation and path-regression tasks.

3) The multitask 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 multitask joint training, we adopt CGC [19] 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 $E_T$ (or task-specific embedding $E_S$). The number of bipartite graphs in modules is a hyperparameter to tune. We use two bipartite graphs as input: one constructed through the interaction matrix ($G_{UI}$) and the other constructed through a meta-path $\Phi$ starting from users and ending with items ($G_{\Phi}$). The node embeddings after propagation of modules ($Z_{S,UI}$), $Z_{S,\Phi}$, $Z_{T,UI}$, and $Z_{T,\Phi}$) are shown as follows:

$$Z_{S,UI} = H(E_S, G_{UI}), \quad Z_{S,\Phi} = H(E_S, G_{\Phi})$$

$$Z_{T,UI} = H(E_T, G_{UI}), \quad Z_{T,\Phi} = H(E_T, G_{\Phi})$$

where $H(\cdot)$ is the encoder function to encode connectivity information into representation learning. The encoder aims to perform convolutional operations on the input graph data, capturing fine-grained node preferences through message passing between adjacent nodes. Since LightGCN [6] has superior performance for recommender systems, we adopt it as the basic structure of our encoder. It is worth noting that LightGCN adopts a layer combination operation to replace the self-loop connection of nodes. However, to avoid the negative effects of outliers in message passing, we remove the layer combination and maintain self-loops. The message passing operation (two propagation layers) with self-loops is defined as follows:

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

$$\hat{D} = D + I$$

$$\hat{A} = A + I$$

B. Embedding Fusion Layer

In the embedding fusion layer, each top-level module absorbs knowledge from shared modules and its task-specific module in the embedding fusion layer. (c) Multitask joint training layer aims to calculate the loss and end-to-end optimization for DUAL. (d) For the contrastive task, data–data pairs are derived from the embeddings of predictive ($Z^{pre}$) and recommendation tasks ($Z^{rec}$).
gating network is formulated as
\[ Z^k = g^k(x)S^k \]  
where \( Z^k \) 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
\[ g^k(x) = \text{softmax}(W^k x) \]
where \( x \in \mathbb{R}^d \) is the input representation, \( W^k \in \mathbb{R}^{(m_k+1)\times d} \) is a parameter matrix, and \( (m_k+1) \) 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 \( (Z_{(I,UI)}) \) or \( Z_{(r,\phi_i)} \) and shared modules \( (Z_{(S,UI)}) \) and \( Z_{(S,\phi)} \)

\[ S^\text{rec} = [Z_{(I,UI)}, Z_{(S,UI)}, Z_{(S,\phi)}]^T \]
\[ S^\text{pre} = [Z_{(r,\phi_i)}, Z_{(S,UI)}, Z_{(S,\phi)}]^T. \]

\section{C. Multitask Joint Training Layer}

The multitask joint training layer first generates two scores indicating the user’s preference for an 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 \( Z^k \) as \( p_{u,v}^k \in \mathbb{R}^d \) and \( q_{u,v}^k \in \mathbb{R}^d \). The scores of user \( u \) for item \( v \) are calculated as follows:

\[ \tilde{R}_{uv} = h^\text{rec} \cdot (p_{u,v}^\text{rec} \odot q_{u,v}^\text{rec}) = \sum_{i=1}^d h_i p_{u,i}^\text{rec} q_{v,i}^\text{rec} \]
\[ \tilde{M}_{uv} = h^\text{pre} \cdot (p_{u,v}^\text{pre} \odot q_{u,v}^\text{pre}) = \sum_{i=1}^d h_i p_{u,i}^\text{pre} q_{v,i}^\text{pre} \]

where \( h^\text{rec} \in \mathbb{R}^d \) and \( h^\text{pre} \in \mathbb{R}^d \) denote the output layers for recommendation and the predictive task, respectively. We construct a recommendation list of candidate items by descending \( \tilde{R}_{uv} \) for the recommendation task.

\textit{1) Loss Function of Recommendation:} We apply the efficient nonsampling learning [2] to optimize our target task—recommendation. It is a state-of-the-art learning method that uses only positive samples to guide model training without negative sampling. It has been confirmed in other studies [1], [2] that it has better performance and efficiency than traditional sampling learning methods. The nonsampling algorithm treats all node pairs that have not interacted as negative samples and incorporates them into the model’s training. Specifically, the recommendation task loss is

\[ L_{\text{rec}} = \sum_{u \in \mathcal{B}} \sum_{v \in \mathcal{I}^+} (c_v^+ - c_v^-) \tilde{R}_{uv}^2 - 2c_v^+ \tilde{R}_{uv} \]
\[ + \sum_{i=1}^d \sum_{j=1}^d \left((\sum_{u \in \mathcal{B}} \sum_{v \in \mathcal{I}} c_v^- q_{v,j} \tilde{R}_{uv} \tilde{R}_{uv}^\text{rec}) (\sum_{v \in \mathcal{I}} c_v^- q_{v,j} \tilde{R}_{uv} \tilde{R}_{uv}^\text{rec}) (h_i \tilde{R}_{uv} \tilde{R}_{uv}^\text{rec}) (h_j \tilde{R}_{uv} \tilde{R}_{uv}^\text{rec})\right) \]

where we denote \( \mathcal{B} \) as a batch of users and \( \mathcal{I} \) as the set of all items. In the derivation of the nonsampling learning, each sample pair \((u, v)\) is given a weight \( c_{uv} \). To facilitate the calculation, we follow the setting in the previous work [2] and set the weight of all positive and negative sample pairs to \( c_v^+ \) and \( c_v^- \), respectively. They are two hyperparameters. \( \mathcal{I}^+ \) is the positive samples set.

\textit{2) Loss Function of Predictive Task:} Fig. 1 shows a strong positive correlation between the interaction and the number of user–item paths induced by meta-paths. The higher the number of path instances between a user and an item, the higher the likelihood of an interaction between them. Predicting the number of user–item connections is beneficial for learning node preference. As a result, we model the number of user–item paths through a regression task.

As for the path-regression predictive task, we use the link-score matrix \( M \) as explicit supervisory signals. We minimize the mean squared error (MSE) between the predictions \( \hat{M} \) and the link-score matrix \( M \)

\[ L_{\text{pre}} = \sum_{u \in \mathcal{B}} \sum_{v \in \mathcal{I}} (M_{uv} - \hat{M}_{uv})^2. \]

Using the link scores to expand supervision can provide more signals to guide the embedded learning than using only history interaction. The predictive task enhances the effectiveness of recommendations by fitting the intimate relationship between nodes.

\textit{3) Loss Function of Contrastive Task:} Influenced by supervised signals, the trained \( Z_{\text{rec}} \) contains the node preference for historical interaction. In contrast, \( Z_{\text{pre}} \), another perspective of node relation, reflects the intimacy degree between nodes following a specific meta-path. Maximizing mutual information allows the intimate relationship between nodes to be further integrated into the \( Z_{\text{rec}} \) for the recommendation task. In addition, through tailored sampling methods, users can be as close as possible to items that strongly semantically relevant but have not seen before. In this way, the model makes recommendations based on semantically relevant items, not just historical ones.

As for the path-guided contrastive task, we maximize the agreement between node representation in \( Z_{\text{rec}} \) and \( Z_{\text{pre}} \). 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. 3, any node in \( Z_{\text{rec}} \) is treated as an anchor \((v_1)\). In \( Z_{\text{pre}} \), the same anchor node \((v_1)\) and nodes whose link score between themselves and the anchor is 1 \((v_1' \text{ and } v_2')\) are positive samples \((v_1, v_1' \text{ and } v_2')\). Nodes whose link score between themselves and the anchor is 0 are naturally regarded as negative samples \((v_1' \text{ and } v_2')\). 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.

For pairs of positive samples, we desire to maximize their consistency under different views. On the contrary, for negative sample pairs, it is necessary to enlarge the divergence between them, forcing them to be farther apart in the latent space. Specifically, we adopt the same form of loss function as SGL [26]—contrastive loss to optimize the path-guided
contrastive task

$$\mathcal{L}_{\text{con}} = \sum_{(i,j) \in Q^+} - \log \frac{\exp(s(Z^\text{rec}_{i}, Z^\text{pre}_{j})/\tau)}{\sum_{(i,j) \in Q} \exp(s(Z^\text{rec}_{i}, Z^\text{pre}_{j})/\tau)}.$$  \hfill (12)

We denote $Q^+$ as the set of positive samples and $Q^-$ as negative samples. The function $s(\cdot, \cdot)$ is designed to measure the distance between two node embeddings in the latent space. Concretely, we employ cosine similarity to calculate the similarity between nodes in a node pair. $\tau$ 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:

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

where $\lambda_{\text{pre}}$ denotes the weight of the predictive task and $\lambda_{\text{con}}$ denotes the weight of the contrastive task. Both $\lambda_{\text{pre}}$ and $\lambda_{\text{con}}$ are hyperparameters. 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 $E_S$ and task-specific node embedding $E_T$). Therefore, our framework retains the low computational complexity.

VI. EXPERIMENTS

In this section, we first display the superiority of our DUAL model by extensive experiments on three public datasets and comparing state-of-the-art recommendation models. Second, we also perform ablation studies on the two auxiliary tasks to demonstrate the efficacy of the hybrid method. Besides, we explore the influence of different meta-paths and encoders on experimental results. Finally, we perform parameter sensitivity experiments to guide the choice of hyperparameter for our DUAL model.

A. Experimental Setup

1) Datasets: We experiment with three publicly accessible datasets from different domains: Last-FM\textsuperscript{1} dataset from the music domain, Yelp\textsuperscript{1} dataset from the business domain, and Douban-book\textsuperscript{1} dataset from the book domain. These datasets are frequently used to validate the effectiveness of previous recommendation models \cite{8}, \cite{31}. We convert ratings in datasets into binary feedback of 0 and 1, which only distinguish whether the user and the item have interacted. Following previous works \cite{36}, we only keep active users and popular items with no less than five interaction records. In addition to the different recommendation scenarios, the datasets also have various data sparsity: 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 shown in Table I.

\textsuperscript{1}https://github.com/librahui/HIN-Datasets-for-Recommendation-and-Network-Embedding

2) Baselines: To verify the superiority of our DUAL model, we compare it with some state-of-the-art models. It is worth noting that BUIR is a predictive model and SGL is a contrastive model.

| Datasets (Density) | Relations (A-B) | Number of A | Number of B |
|-------------------|-----------------|-------------|-------------|
| Last-FM (1.36%)   | User-User       | 18,802      | 1,892       |
|                   | User-Artist     | 92,834      | 1,892       |
|                   | Artist-Artist   | 153,399     | 17,632      |
| Yelp (0.54%)      | User-User       | 158,590     | 16,239      |
|                   | User-Business   | 198,397     | 16,239      |
|                   | Business-Category | 40,009    | 14,284      |
| Douban-Book (0.33%) | User-User     | 169,130     | 12,748      |
|                   | User-Book       | 792,026     | 13,024      |
|                   | Book-Author     | 21,905      | 22,347      |

TABLE I Statistics of the Datasets

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
1) **NeuMF** [7]: A classic recommendation framework that replaces the embedding inner product through a neural network. Each feature extraction module holds its independent representation.

2) **SLIMEelastic** [15]: A classic method in the top-N recommendation. It improves recommendation accuracy through an auxiliary sparse coefficient matrix learned during model training.

3) **NGCF** [23]: A graph-based collaborative filtering method. It can aggregate node embeddings in a graph structure constructed from interactive data, explicitly putting collaborative filtering signals into the embedding propagation process.

4) **LINE** [20]: A network embedding model. It can map all nodes to a low-dimensional embedding while maintaining the original network structure as much as possible. The method can handle large-scale and various complex data.

5) **DGCF** [24]: A graph-based collaborative filtering method, which can mine and refine various intentions of users and learn the critical factors of interaction between the user and the item.

6) **LightGCN** [6]: An excellent graph-based method that removes damage recommendation’s complex operation during graph convolution. It achieves double improvement in capability and efficiency of recommendation by simplifying NGCF.

7) **BIUR** [10]: A framework for collaborative filtering. It only uses positive samples to guide model training and alleviates the problem of data sparsity through random data augmentation. This method is a predictive SSL model.

8) **SGL** [26]: A framework based on SSL, which performs data augmentation on the input bipartite graph with a dropout of nodes and edges. Then, it draws on the idea of contrastive learning to construct an SSL task.

3) **Evaluation Metrics:** In order to more intuitively demonstrate the superiority of the DUAL model, we use two typical evaluation metrics, Recall@K and NDCG@K, to evaluate the model’s performance. These two evaluation metrics are widely applied in recommendation methods [6], [23], [24]. Recall focuses on the coverage of positive samples in the recommendation list, and NDCG evaluates sorting accuracy.

4) **Implementation Details:** We randomly divide each public dataset and use 80% of the data for model training, 10% as the valid set, and the rest as the test set. The valid and test data are, respectively, used to determine model parameters and evaluate the final model. The parameter settings of baselines are the same as those in their paper and then tuned for optimal performance. For the fairness of the comparison, we set the embedding dimension of all models to 256. In addition, we set the patience of the early stopping method to 20, i.e., models stop training if the performance does not improve for 20 consecutive epochs. For models employing the sampling strategy, we uniformly set the number of negative samples to 5.

For the DUAL model, we randomly initialize parameters and optimize the model with Adam [9]. We set the learning rate to 0.001, and the dropout ratio was set to 0.3 to prevent overfitting. The weight of nonzero instances $c^+_v$ is set to 1 for all datasets. The negative weight $c^-_v$ is set to 0.15, 0.2, and 0.25 for Last-FM, Yelp, and Douban-book, respectively. We set the weight of the path-regression task $\lambda_{pre}$ as 0.3 for Last-FM and 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 $\lambda_{con}$ 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 $\lambda_{con}$ is set to $5e^{-5}$ 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 results of DUAL and baselines are listed in Table II. In experiments, we set $K$ of Recall@K and NDCG@K as 5, 10, 15, and 20 to evaluate the DUAL and baselines. According to the results, the following results can be analyzed.

First, the DUAL outperforms all baselines on three public datasets, achieving the best performance. The average improvement of our DUAL model over the best baseline is 3.96% on the Last-FM dataset, 8.02% on the Yelp dataset, and 10.74% on the Douban-book dataset, which validates the superiority of the DUAL 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.

We believe that the superior performance of DUAL comes from the following three aspects.

1) We adopt a multitask sharing structure, which can explicitly separate shared modules and task-specific modules and extract and separate semantic knowledge reflected by social relationships and item categories, though multitasking joint representation learning improves model performance.

2) DUAL can accurately capture the strong correlation between the number of path instances following the specific meta-path and user-item interactions.

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, LightGCN, SGL, and DUAL outperform NGCF and DGCF. This conclusion is consistent with previous work [6], which indicates the strong performance of LightGCN in the recommendation. LightGCN has such strong performance because it is the progressive graph-based CF model, which designs a light graph convolution to ease the training difficulty and pursue better generative ability.
Third, SSL methods (BIUR, SGL, and DUAL) generally perform better than supervised methods. 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 (BIUR) or contrastive task (SGL). It verifies the effectiveness of both types of auxiliary tasks in modeling user preferences. In addition, the self-supervised tasks of BIUR and SGL models, which are universal self-supervised tasks, originate from computer vision and graph learning, respectively. Our model DUAL outperforms these two baselines, proving that self-supervised task closely related to recommendation is superior to general self-supervised tasks. Regarding the poor performance of the self-supervised model BIUR on the Yelp and Douban-book datasets, we speculate that because BIUR is only trained with positive samples, it does not work well on extremely sparse datasets.

C. Ablation Study

In this section, we conduct ablation studies on the path-regression task and path-guided contrastive task to demonstrate the efficacy of our 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.

1) DUAL-PC: The variant model of DUAL without predictive and contrastive tasks.

2) DUAL-C: The variant model of DUAL without contrastive task.

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

D. Impacts of Different Meta-Paths

To further evaluate our DUAL model, we conduct an experiment on impacts of using different meta-paths. As shown
in Fig. 4, the DUAL with different meta-paths performs better than the best performing baseline SGL, which means that combining meta-paths in the framework is effective for improving recommendations. The 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 to facilitate various graph-based recommendation methods. We evaluate the DUAL model with five graph neural networks: GCN [5], GAT [22], TAGCN [4], SGC [25], and LightGCN [6]. As we expected, the results in Table IV demonstrate that the effectiveness of the framework 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. This 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. Hyperparameter Sensitivity

To guide the parameter choice of the DUAL model, we perform hyperparameter studies of the result of DUAL. Specifically, the dataset used to evaluate the model performance is Last-FM, and LightGCN is applied as the encoder of DUAL. We test the result changes of the DUAL model concerning hyperparameters on the number of layers and 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.](image)

For the number of convolutional layers of DUAL, we choose its range in the range of [1, 2, 3, 4, 5] and plot the results in Fig. 5. The result demonstrates that the effect of DUAL is relatively more susceptible to the number of layers than the link score of negative samples. As can be seen from Fig. 5, the model performance is improved by increasing the number of convolutional layers from 1 to 2. Usually, two propagation layers are adequate to capture heterogeneous signals. DUAL suffers from the oversmoothed 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 shown in Fig. 5, DUAL is susceptible to the link score of negative samples, and we need to choose it carefully for the best result. Usually, small values of link score can lead to ideal results. 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 framework is effective.

VII. CONCLUSION AND FUTURE WORK

In this work, we propose a self-supervised 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. In addition, our proposed model DUAL effectively combines two auxiliary tasks while avoiding negative transfer. Experimental results on public datasets demonstrate that DUAL exceeds state-of-the-art supervised and self-supervised recommendation methods.

We will study how to develop a simplified SSL framework with multiple meta-paths in the future. Besides, we will also consider applying the framework to more complicated recommendation scenarios, such as bundle or group recommendation.

REFERENCES

[1] C. Chen et al., “Graph heterogeneous multi-relational recommendation,” in Proc. AAAI, vol. 35, no. 5, 2021, pp. 3958–3966.
[2] C. Chen, M. Zhang, Y. Zhang, Y. Liu, and S. Ma, “Efficient neural matrix factorization without sampling for recommendation,” ACM Trans. Inf. Syst., vol. 38, no. 2, pp. 1–14, 2020.
[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,” IEEE Trans. Computat. Social Syst., vol. 8, no. 1, pp. 86–98, Feb. 2021.
[4] J. Du, S. Zhang, G. Wu, J. M. Moura, and S. Kar, “Topology adaptive graph convolutional networks,” 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 Proc. 34th Int. Conf. Mach. Learn., vol. 70, 2017, pp. 1263–1272.
[6] 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. Retr., Jul. 2020, pp. 639–648.
[7] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, “Neural collaborative filtering,” in Proc. 26th Int. Conf. World Wide Web (WWW), 2017, pp. 173–182.
[8] 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 Proc. NIPS, 2020, pp. 1–12.
[9] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. ICLR, 2015, pp. 1–15.
[10] D. Lee, S. Kang, H. Ju, C. Park, and H. Yu, “Bootstrapping user and item representations for one-class collaborative filtering,” in Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr., Jul. 2021, pp. 1513–1522.
[11] J. Li, L. Wu, R. Hong, and J. Hou, “Random walk based representation learning and prediction on social networking services,” Inf. Sci., vol. 549, pp. 328–346, Mar. 2021.
[12] J. Li, L. Wu, R. Hong, K. Zhang, Y. Ge, and Y. Li, “A joint neural model for user behavior prediction on social networking platforms,” ACM Trans. Intell. Syst. Technol., vol. 11, no. 6, pp. 1–25, Dec. 2020.
[13] J. Ma, C. Zhou, H. Yang, P. Cui, X. Wang, and W. Zhu, “Disentangled self-supervision in sequential recommenders,” in Proc. SIGKDD, 2020, pp. 483–491.
[14] 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 Proc. SIGKDD, 2018, pp. 1930–1938.
[15] X. Ning and G. Karypis, “SLIM: Sparse linear methods for top-N recommender systems,” in Proc. IEEE 11th Int. Conf. Data Mining, Dec. 2011, pp. 497–506.
**Min Gao** (Member, IEEE) received the M.S. and Ph.D. degrees in computer science from Chongqing University, Chongqing, China, in 2005 and 2010, respectively. She was a Visiting Researcher with the University of Reading, Reading, U.K., and Arizona State University, Tempe, AZ, USA. She is currently an Associate Professor with the School of Big Data and Software Engineering, Chongqing University. 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. Her research interests include recommendation systems, service computing, and data mining. She has published over 40 refereed journal articles and conference papers in these areas. Dr. Gao is a member of China Computer Federation (CCF).

**Junliang Yu** received the B.S. and M.S. degrees in software engineering from Chongqing University, Chongqing, China, in 2013 and 2018, respectively. He is currently pursuing the Ph.D. degree with the School of Information Technology and Electrical Engineering, The University of Queensland, Saint Lucia, Queensland, Australia. His research interests include recommender systems and anomaly detection.

**Zongwei Wang** received the B.S. and M.S. degrees in software engineering from Chongqing University, Chongqing, China, in 2018 and 2021, respectively. His research interests include recommender systems and anomaly detection.

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

**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. She is currently a Professor with the College of Mechanical and Vehicle Engineering, Chongqing University. Her research interests include modern logistics and supply chain management, industrial Internet, and service science and engineering.