Information Sciences 703 (2025) 121946

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

GMR-Rec: Graph mutual regularization learning for multi-domain recommendation

Yifan Wang [a], Yangzi Yang [b], Shuai Li [c], Yutao Xie [a], Zhiping Xiao [d,*], Ming Zhang [c,*], Wei Ju [e,*]

[a] School of Information Technology & Management, University of International Business and Economics, Beijing, 100029, China
[b] School of Economics and Management, Beihang University, Beijing, 100191, China
[c] School of Computer Science, National Key Laboratory for Multimedia Information Processing, Peking University, Beijing, 100871, China
[d] Computer Science & Engineering, University of Washington, Seattle, 98195, USA
[e] College of Computer Science, Sichuan University, Chengdu, 610064, China

ARTICLE INFO

Keywords:
Multi-domain recommendation
Mutual regularization learning
Graph neural networks
Collaborative filtering

ABSTRACT

Multi-domain recommender systems are becoming increasingly significant, as they can alleviate the sparsity challenge and cold-start problem within a single domain by transferring knowledge from related domains in a collective manner. However, existing methods primarily concentrate on the process of sharing or mapping the features of the same users across different domains to facilitate knowledge transfer. Since the user-item interactions can be naturally formulated as bipartite graphs, transferring knowledge via message passing throughout domains would be a more straightforward approach. Moreover, the existing approaches generally pay more attention to modeling the common interests of users, leaving behind the under-explored domain-specific interests. In this paper, we introduce a novel framework, called GMR-Rec, for the multi-domain recommendation, which explicitly transfers knowledge across various domains. Specifically, both domain-shared and domain-specific graphs are constructed using historical user-item interactions, with the parallel graph neural network employed for each of them. Then, mutual regularization strategies are proposed to distinguish domain-specific user interests while preserving common user interests shared across domains. Experimental results on the four real-world datasets show that our model achieves an average improvement of 1.24%, 2.90%, 5.07% and 3.17% in HR@10, and 3.05%, 4.24%, 6.38% and 3.99% in NDCG@10 compared to the state-of-the-art baseline.

1. Introduction

As Internet services develop in a rapid manner, our daily life is unprecedentedly linked to online services, sparking a vast increase in the production of online information. As a result, the personalized recommender system has become a prevalent service for routing users to the preferred items among millions of alternatives. Collaborative Filtering (CF), which models connections among users as well as dependencies among items, has achieved remarkable success in recommender systems.

* Corresponding authors.
E-mail addresses: yifanwang@uibe.edu.cn (Y. Wang), ymyang@buaa.edu.cn (Y. Yang), lishuai@stu.pku.edu.cn (S. Li), yutaoxie@uibe.edu.cn (Y. Xie), patxiao@uw.edu (Z. Xiao), mzhang_cs@pku.edu.cn (M. Zhang), juwei@scu.edu.cn (W. Ju).

https://doi.org/10.1016/j.ins.2025.121946
Received 15 April 2024; Received in revised form 2 February 2025; Accepted 2 February 2025

Available online 5 February 2025

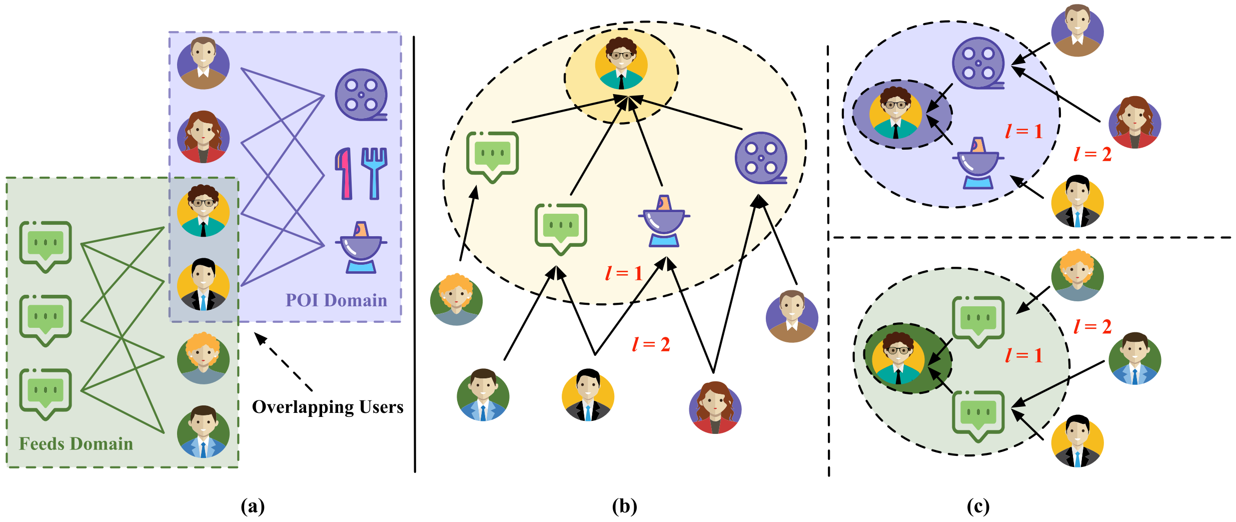


Fig. 1. An illustration in Dianping APP. (a) User-item interactions in Feeds and POI domains; (b) High-order connectivities in the shared domain; (c) High-order connectivities in the specific domains.

The key point behind CF models involves learning informative representations (i.e., embeddings) of users and items. Traditional CF methods (i.e., Matrix Factorization (MF) [1]) derive these embeddings through historical user-item interactions solely and make predictions based on the embedding similarity. However, these methods confront challenges arising from data sparsity and the cold-start problem, notably impacting the learned representation of users with few interactions and leading to poor performance. Fortunately, users often have interactions in multiple domains in the real world. Therefore, the integration of auxiliary data from these additional domains may play a crucial role in enabling the recommender system to address issues related to the data sparsity and the cold-start problem.

Multi-domain recommendation endeavors to enhance recommendation performance in each domain by leveraging user interests from other auxiliary domains. For example, CodeBook Transfer (CBT) [2] converts the auxiliary rating matrix into a cluster-level representation of rating patterns, referred to as a codebook. Then it transfers the codebook into the target domain, and reconstructs the target rating matrix. Different from the two-stage migration utilized by CBT, Collective Matrix Factorization (CMF) [3] conduct factorization on the rating matrices from various domains, and directly shares the latent features of common users or items to facilitate knowledge transfer. Meanwhile, a few follow-up works are proposed to improve this approach [4,5]. However, existing approaches mostly focus on sharing/mapping user representations to implicitly transfer knowledge across domains. Intuitively, we can formulate the user-item interaction records as a bipartite graph. By integrating the interactions from multiple domains as one shared graph, we can build a better model to capture the user-item relations by considering high-order connectivity from different domains and transferring knowledge explicitly through message passing between these relations.

In recent studies, it has been demonstrated that using the user-item graph structure can effectively alleviate the data sparsity problem and the cold-start problem [6,7]. As an example from the online review APP Meituan Dianping shown in Fig. 1(a), users spend leisure time browsing reviews through continuously updated feeds. However, when they wish to directly access a Point of Interest (POI), they still resort to keyword-based searches. By linking the interaction graph of the POI domain to the Feeds domain, it becomes possible to learn and leverage common knowledge across these domains, enhancing the user experience by creating a more seamless link between browsing and searching activities. Since Graph Neural Networks (GNNs) achieved great success in extracting features from non-Euclidean spaces, recent works have employed GNN-based information propagation strategies to learn the embeddings on the constructed user-item bipartite graph. For example, NGCF [6] propagates in the same way as GNNs (conducting feature transformation, neighborhood aggregation, and nonlinear activation) to refine the embeddings of users and items. LightGCN [7] simplifies GNNs' design by propagating the embeddings on the user-item bipartite graph in a linear manner, split the nonlinear activation function and feature transformation from the information propagation process, under the assumption that these steps bring too much burden to collaborative filtering.

Despite the effectiveness of these multi-domain and GNN-based recommendation methods, there are a few remaining challenges. **First**, most methods only focus on domain-shared features, ignoring domain-specific features. For example, as shown in Fig. 1(b), in the POI domain, the target user searches for both cinema and hotpot-related keywords, whereas in the Feeds domain, the user focuses solely on food reviews. Since cinema is not this user's interest in the Feeds domain, it should not be used for recommendation. Thus, directly sharing/mapping user interests across domains may lead to unsatisfactory results. **Second**, existing methods struggle to effectively capture the distinct characteristics of interaction records. These records can naturally be represented as a heterogeneous bipartite graph, consisting of two different types of entities: users and items. However, most GNN-based recommendation methods overlook the heterogeneity of node types and propagate embeddings in a recursive manner from distant to nearby neighbors, resulting in suboptimal and inefficient feature extraction, particularly in multi-domain recommendation scenario. **Third**, since neighborhood embeddings are recursively aggregated from each layer, the high-order feature interactions across layers are implicitly modeled, leading to non-robustness and low interpretability when calculating matching scores between user and item.

Toward this end, in this paper, we propose **GMR-Rec**, which is a novel **G**raph **M**utual **R**egularization learning framework designed to address the above-mentioned challenges in multi-domain **Rec**ommendation settings. On the one hand, we form a training cohort with independent models from each domain so as to capture the corresponding features of the users. Both the domain-shared user interests and the domain-specific ones can be distinguished through our proposed mutual learning strategy. On the other hand, the information propagation in GNNs can be seen as a neighborhood embedding combination process. It is very natural to design a new structure of GNN based on the attention mechanism, which can explicitly detail how information propagates across layers, enabling it to distinguish between types of neighbors based on their layer depth. In particular, for each domain, we first construct the domain-specific interaction graph accordingly, and link the individual graphs together to form a domain-shared graph that spans all domains. Then, instead of transferring the common knowledge among multiple domains by sharing or mapping their embeddings, we propose a novel parallel GNN approach to aggregate the neighbor embeddings under different hops settings independently, and combine them using a standard multi-head self-attention mechanism [8]. The process can be regarded as another way of embedding propagation where the user interests, both the domain-shared and the domain-specific ones, are refined based on the aggregated neighborhood information from different hops. On this basis, mutual regularization strategies among different domains are proposed to enforce the users' specific interests in different domains to become as distinguishable from each other as possible, while capturing the common interests that are shared across these domains. Finally, both the domain-shared user interests and the domain-specific user interests are integrated as the final user interest for user preference prediction.

In conclusion, our main contributions are as follows:

- We proposed a graph mutual regularization approach called GMR-Rec for recommendation tasks under multi-domain settings. It improves the performance of recommendation tasks on each domain, via training a cohort of GNNs in other domains collaboratively. The approach encourages domain-specific features to be distinguished from each other while simultaneously ensuring that these distinct features remain closely aligned with the domain-shared feature.
- Based on GMR-Rec, we proposed to employ a novel parallel GNN framework, which exploits the heterogeneity of the bipartite graph representation of user-item relations, by independently aggregating information from the neighborhood at different hops and combining them with a new attention mechanism.
- Extensive experiments have been conducted on four real-world datasets, and the results showed that our model outperforms the existing state-of-the-art methods. Additionally, we have also conducted some further experiments exploring the effectiveness of each component, the impact of hyperparameters and case studies to explore the effectiveness of our framework.

Our paper is structured as follows. In Section 2, we conclude the existing related works. In Section 3, the preliminaries and problem definitions of our work are introduced. Then, in Section 4, we detail each component of our GMR-Rec. Next, in Section 5, we evaluate the effectiveness of GMR-Rec on four different real-world datasets. Followed by Section 6 that concludes the paper.

2. Related work

We introduce three lines of the existing related research works: (1) Multi-domain Recommendation, (2) GNN-based Recommendation and (3) Knowledge Distillation and Mutual Learning.

2.1. Multi-domain recommendation

Data sparsity and cold start problems are the main challenges in recommender systems [9,10]. To address these problems, multi-domain recommendation uses other relevant domains as auxiliary information to transfer corresponding knowledge, especially if the number of other domains is greater than one [11–14]. A certain category of methods employs Matrix Factorization (MF) within each domain and seeks to establish connections between these domains [2–4]. For example, CBT [2] builds a cluster-level pattern matrix named codebook to represent the dense rating matrix and then shares the codebook of auxiliary domains to the target domain. CMF [3] factorizes the rating matrices across multiple domains jointly, and it shares the user latent factors among these domains. CDFM [15] extends factorization machines (FM) and treats user's interactions in other domains as context for target domain recommendation. Compared with these shallow multi-domain models, since recent machine learning especially deep learning possesses strong feature extraction capabilities and the ability to learn complex patterns [16,17], some existing deep learning models have been introduced to improve knowledge transfer across domains. For example, EMCDR [18] explicitly maps the representations of the same user from different domains via a multi-layer perceptron (MLP). CoNet [19] introduces cross-connections in the hidden layers of networks to achieve the effect of dual knowledge transfer between domains. DDTCDR [5] utilizes autoencoder to extract the features of users and items respectively, employing a latent orthogonal mapping to maintain the similarity of user interests in different domains. UniCDR [20] takes one step further, providing a unified framework to solve multi-domain recommendation problems by learning domain-shared and domain-specific user interests. However, the above-mentioned methods mainly focus on sharing/mapping the embeddings for common users (or items), neglecting the explicit transfer of knowledge across different domains.

2.2. GNNs-based recommendation

Recently, GNNs [21–24] have been widely known as a promising type of models that are typically good at capturing dependencies among graph nodes. Based on GNNs' capability, some works model user-item relations as graphs, and then utilize GNNs to capture the higher-order relationships in this graph, so as to improve the recommender systems' performances [25,6,7]. PinSAGE [25] obtains item embeddings through GraphSAGE [22] on the item-item graph and then performs item2item recall based on the item embeddings. NGCF [6] stacks several GCN layers together, so as to learn higher-order connectivity signals from all the user-item pairs. LightGCN [7] made some simplifications based on the vanilla NGCF, removing the feature transformation operations and the activation functions, which were shown to have no positive impact on the effectiveness of CF. Some other works integrate some additional information and, therefore, build heterogeneous versions of user-item graphs, utilizing GNNs to obtain embeddings of users or items [26,27]. Compared to the traditional multi-domain recommendation methods, the key idea of GNNs is to use graph-structure aware neural networks to propagate the node features from neighbors to target nodes, which is suitable for the multi-domain recommendation. There are some works [28–30] that leverage GNNs to transfer knowledge between different domains. However, these methods tend to focus on user interests that are shared across domains, while neglecting the unique user interests specific to each domain.

2.3. Knowledge distillation and mutual learning

Recently, since Knowledge Distillation (KD) has shown its capability in model compression [31–33], it has been widely applied to image recognition [34], natural language processing (NLP) [35], time series prediction [36,37] and graph representation learning [38], where the student network trained with KD has comparable performance to the teacher model but with lower latency due to its smaller size. For example, DTCM [36] proposes a targeted and offline distillation method for dual-network-based student and teacher models, facilitating effective knowledge transfer for multivariate time series classification. CapMatch [37] leverages feature-based KD to effectively transfer knowledge, enabling the model to capture both local and global patterns in human activity recognition data within a semi-supervised learning framework. For recommendation, most works employ KD to transfer some specific knowledge from the other auxiliary models to improve the recommender performance or its interpretability [39–41]. Mutual learning [42,43], which can be regarded as a special case of KD, is an ensemble of many student networks that learn from each other via the distillation loss. Partly inspired by mutual learning, our work, to the best of our knowledge, is the first work that attempts to solve the multi-domain recommendation problem via exploiting graph mutual regularization learning.

3. Preliminaries

We first provide the formal definition of the multi-domain recommendation problem. Then we introduce the limitation associated with iterative GNN frameworks and explain how parallel GNN frameworks offer a different approach to addressing the challenge.

3.1. Problem definition

We consider several domains $\{D_1, \ldots, D_K\}$. The users in each domain are partly shared, and we denote the entire set of the users as $\mathcal{U} = \{u_1, u_2, \ldots, u_M\}$ (whose size is $M = |\mathcal{U}|$), and item set at each domain as $\mathcal{V}_k = \{v_{k_1}, v_{k_2}, \ldots, v_{k_{N_k}}\}$ (whose size is $N_k = |\mathcal{V}_k|$). Based on these definitions, we define a user-item interaction matrix $Y_k \in \{0,1\}^{M*N_k}$ for each domain k, where for its entry at u, v_k:

$$y_{u,v_k} = \begin{cases} 1 & \text{if user } u \text{ engages with item } v_k \text{ in domain } k \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Based on the interaction matrix Y_k, we construct $\mathcal{G}_{s_k}$, a domain-specific bipartite graph depicting the user-item relations of domain k. Meanwhile, we link the set of graphs of all domains $\{\mathcal{G}_{s_1}, \mathcal{G}_{s_2}, \ldots, \mathcal{G}_{s_K}\}$ by using the overlapped users as anchors, and then construct a domain-shared graph $\mathcal{G}_c$ accordingly. Our goal is to learn the low-dimensional representations of user u and item v_k for prediction function $\hat{y}_{u,v_k} = \mathcal{F}(u, v_k | \Theta, \{\mathcal{G}_{s_1}, \ldots, \mathcal{G}_{s_K}, \mathcal{G}_c\})$, where we use $\hat{y}_{u,v_k}$ to denote the probability of user u getting engage with item v_k, and use Θ to denote the set of parameters in the model that we used to implement the prediction function $\mathcal{F}$.

3.2. Iterative and parallel GNNs framework

A general iterative GNN framework stacks multiple GNN layers to update the node embeddings iteratively. Given that $H_t^{(l)}$ represents the embedding of a target node t at the (l)-th layer of GNN, the iterative updating process from the $(l-1)$-th to the (l)-th layer can be formally depicted as:

$$H_t^{(l)} = \mathbf{Combine}\Big(H_t^{(l-1)}; \underset{\forall s \in N(t)}{\mathbf{Aggregate}}(H_s^{(l-1)})\Big), \tag{2}$$

where we use $N(t)$ to denote the collection of all source nodes available for target node t. $\mathbf{Combine}(\cdot)$ and $\mathbf{Aggregate}(\cdot)$ are the two basic operations that recursively combine and aggregate information from neighbors.

However, the iterative GNN framework needs each successive layer to use the outputs of its previous layer as inputs, which is less effective and inefficient for multi-domain recommendation. Instead, a parallel GNN framework aggregates each hop of neighborhood

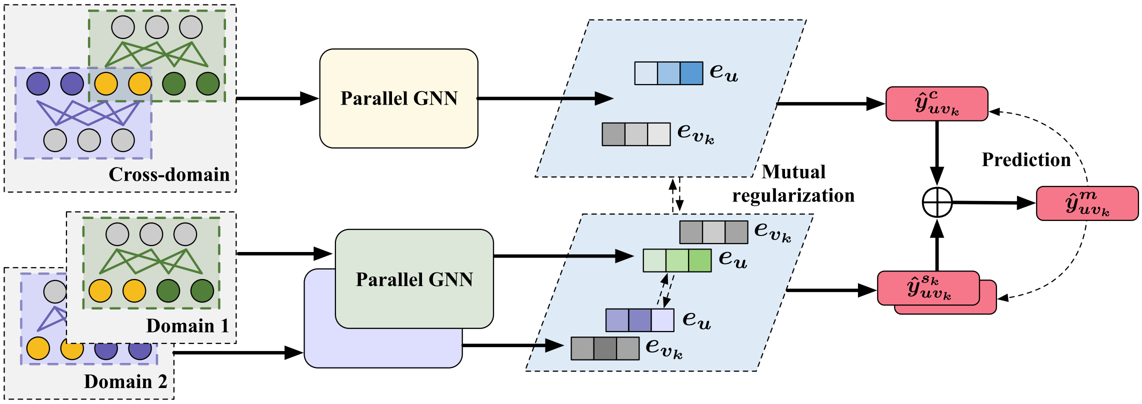


Fig. 2. The framework of GMR-Rec, where the grey circles refer to the items, while items may come from two different domains. The yellow circles are the overlapping users between the two domains. We use multiple parallel GNNs (represented by different colors) to extract the domain-shared and the domain-specific features, respectively. Mutual regularization strategies are performed to achieve knowledge transfer among domains for multi-domain recommendation.

embeddings independently and ensemble embeddings of the node itself as well as each hop of aggregated neighborhood embeddings to avoid recursively propagating.

4. The proposed model

We first introduce the overall framework of our proposed GMR-Rec, which can transfer knowledge among domains by training collaboratively with a cohort of GNNs. Then we introduce each component in detail.

4.1. Overview

The core idea is to identify the domain-shared and domain-specific user interests explicitly, via training recommendation models across and within the individual domains. And the learned user interests complement each other to enrich the final recommendation. Fig. 2 shows the GMR-Rec framework, which consists of an ensemble of parallel GNN models, a mutual regularization framework and an ensemble of prediction layers. Given constructed user-item interaction graphs of multiple domains, our model links them into a whole graph by the overlapping users and utilizes a series of proposed parallel GNNs to extract the user interest. Then, domain-shared and domain-specific user interests are distinguished using the proposed mutual regularization approach. Finally, the integrated user interest is used to predict the matching scores, ensuring a comprehensive recommendation.

4.2. Parallel GNN

Aiming to capture the heterogeneity within the user-item interaction graph $\mathcal{G}$, a parallel graph neural network mechanism (parallel GNN) is applied to model the influences of neighbor nodes with different types and distances to the target node. For a target node denoted as node t, we sample its neighbors within L-hops, and update the embedding of t by independently aggregating each of the l-hop neighborhoods $N_t^{(l)}$, where $l \in \{1, 2, \ldots, L\}$.

4.2.1. Parallel neighborhood aggregation

As shown in Fig. 3, instead of recursively updating the node embeddings in each layer (e.g., (l)-th later) with its neighbors' embeddings from the previous layer (e.g., $(l-1)$-th later), we directly aggregate multiple neighborhood embeddings from neighbors at l-hop for $l \in \{1, \ldots, L\}$ in parallel. We leverage the attention mechanism [8] to aggregate messages independently at each hop according to its node type. Specifically, we first apply a type-specific transformation on user and item embeddings, mapping both into the same latent factor space. Next, for the target node t, we calculate the attention weight $a_{ts}^{(l)}$ on each l-hop neighbor $s \in N_t^{(l)}$ and weighted summing their projected features, defined as:

$$\begin{aligned} \alpha_{ts}^{(l)} &= \underset{\forall s \in N_t^{(l)}}{\mathrm{Softmax}}\Big(\mathrm{ReLU}\big(a_l^{\mathrm{T}}(e_t \parallel e_s)\big)\Big), \\ z_t^{(l)} &= \sum_{\forall s \in N_t^{(l)}} a_{ts}^{(l)} \cdot e_s, \end{aligned} \tag{3}$$

where $e_t, e_s \in \mathbb{R}^d$ are the projected embedding of target and source nodes respectively, $a_l \in \mathbb{R}^{2d}$ is the parameterized attention vector for l-hop neighbors, $\parallel$ denotes the concatenation operation. We extend this attention mechanism to multiple heads, which separately repeat the attention H_α times and concatenate the learned features as output:

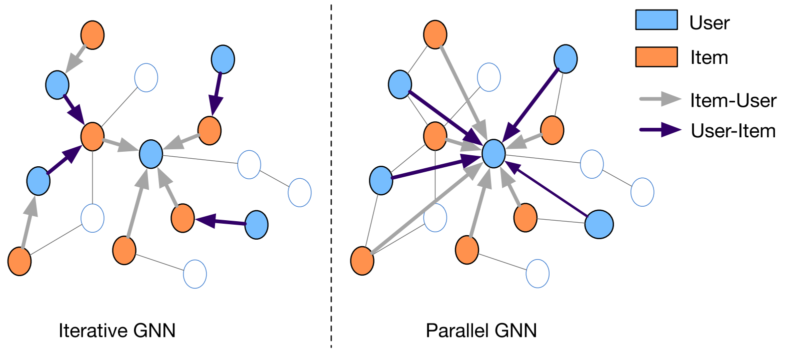


Fig. 3. An illustration of Iterative and Parallel GNN, where Iterative GNN leverages each successive layer by using the outputs of its previous layer as inputs, while Parallel GNN aggregates information from each layer independently to avoid recursive propagation.

$$z_t^{(l)} = \Big(\Big\|_{h=1}^{H_\alpha} \sum_{\forall s \in N_t^{(l)}} \big[\alpha_{ts}^{(l)}\big]_h \cdot e_s \Big) W_O^{(l)}, \tag{4}$$

where we use $\big[\alpha_{ts}^{(l)}\big]_h$ to represent the attention weight at the h-th attention head, and $W_O^{(l)} \in \mathbb{R}^{H_\alpha d \times d}$ is the parametric weight matrix for l-hop neighbors.

4.2.2. Cross-hop neighborhood propagation

After aggregating the neighborhood features within each hop, it becomes crucial to propagate among these embeddings. Instead of recursively propagating messages to the target node, we develop a hop-wise version variant of the multi-head self-attention layer, which can directly propagate different hops of aggregated neighborhood embeddings. Specifically, for target node t, we have a set including $L+1$ latent vectors: $\{z_t^{(0)}, z_t^{(1)} \dots z_t^{(L)}\}$, where $z_t^{(0)} = e_t$. Following the Transformer framework [8], we sum the position embedding to each latent vector so that the model can identify which hops of neighbors (including the target node itself) it is dealing with. Then, we pack all these vectors into a matrix Z_t and map it into the Query, Key, and Value matrix, respectively. We apply H_β attention heads to produce the propagated representations ATT-head$_h$ which are concatenated and then projected again as output MH(Z_t), which can be defined as:

$$\begin{aligned} \text{ATT-head}_h &= \text{Softmax}\Big(\frac{(Z_t W_Q^h)(Z_t W_K^h)^{\mathrm{T}}}{\sqrt{d}} \Big), \\ \text{MH}(Z_t) &= \Big(\Big\|_{h=1}^{H_\beta} \text{ATT-head}_h Z_t W_V^h \Big) W_O, \end{aligned} \tag{5}$$

where the projections are parameter matrices $W_Q^h, W_K^h, W_V^h \in \mathbb{R}^{d \times d}$ and $W_O \in \mathbb{R}^{H_\beta d \times d}$. We use dropout and Layer Normalization (LN) to MH(Z_t) to avoid overfitting and stabilize training:

$$Z_t' = \text{LN}\big(Z_t + \text{Dropout}(\text{MH}(Z_t))\big). \tag{6}$$

And to endow the propagation process with non-linearity, we employ a position-wise FFN (i.e., Feed-Forward Network) module on top of the output Z_t' that comes from the self-attention layer, defined as follows:

$$\hat{Z}_t = \text{LN}\big(Z_t' + \text{Dropout}(\text{ReLU}(Z_t' W_{F_1} + b_{F_1}) W_{F_2} + b_{F_2})\big), \tag{7}$$

where FFN's learnable parameters are $W_{F_1}, W_{F_2} \in \mathbb{R}^{d \times d}$ and $b_{F_1}, b_{F_2} \in \mathbb{R}^d$. We can stack multiple such self-attention layers one after another, taking the previous layer's output as the next layer's input. By doing so, we can learn high-order interaction features between different hops of neighborhoods.

4.3. Recommendation

Treating a user u as the target node, the corresponding output from the parallel GNNs can be regarded as a set of refined embeddings $\{\hat{z}_u^{(0)}, \hat{z}_u^{(1)} \dots \hat{z}_u^{(L)}\}$, representing the user features learned in different hops. Though there could be much more complicated ways of combining these vectors together, an effective version of the final embedding of user u is simply taking the concatenation of all these vectors. Meanwhile, we consider an item as the target node and concatenate its features $\{\hat{z}_v^{(0)}, \hat{z}_v^{(1)} \dots \hat{z}_v^{(L)}\}$ learned in different hops in the same way:

$$\hat{e}_u = \hat{z}_u^{(0)} \parallel \cdots \parallel \hat{z}_u^{(L)}, \hat{e}_v = \hat{z}_v^{(0)} \parallel \cdots \parallel \hat{z}_v^{(L)}. \tag{8}$$

By implementing this approach, we can enrich the initial node embeddings by capturing the connectivities across different hops of neighborhood. Finally, we pass the concatenation of the users' and items' final representations through an MLP, and take the output as the estimated matching score:

$$\hat{y}_{uv} = w_2^{\mathrm{T}} \mathrm{ReLU}\big((\hat{e}_u \parallel \hat{e}_v) W_1 + b_1\big), \tag{9}$$

where we have learnable parameters $W_1 \in \mathbb{R}^{Ld\times d}$ and $b_1 \in \mathbb{R}^d, w_2 \in \mathbb{R}^d$. The model with parameters set Θ is trained with an objective with negative sampling, which is defined as:

$$\mathcal{L}_{\Theta} = \sum_{(u,v)\in \mathcal{Y}\cup\mathcal{Y}^-} \mathcal{L}_{\mathrm{ce}}(y_{uv}, \hat{y}_{uv}), \tag{10}$$

where y_{uv} represents the label of the corresponding user-item interaction relation, as was defined in Equation (1). We consider both the positive items $\mathcal{Y}$ and the negative items $\mathcal{Y}^-$. The negative items are sampled uniformly at random from the unobserved list of items. We define our objective $\mathcal{L}_{ce}$ as a binary cross-entropy loss, computed as:

$$\mathcal{L}_{\mathrm{ce}}(\hat{y}, y) = -y\log\hat{y} - (1-y)\log(1-\hat{y}). \tag{11}$$

4.4. Mutual regularization strategy

To achieve more effective inter-domain knowledge transferring, we propose two mutual regularization strategies together with a gated interest ensemble method for the learning process of parallel GNN models.

4.4.1. Domain-shared interest regularization

Our domain-shared interest regularization strategy aims to make the user interests in each domain as close as possible to the common user interests extracted across domains. On the one hand, domain-shared graph $\mathcal{G}_c$ contains all the items from all domains that a user $u \in \mathcal{U}$ has ever interacted with. Thus, the extracted user embedding $\hat{e}_u^c$ in $\mathcal{G}_c$ reflects u's domain-shared interest. On the other hand, domain-specific graph $\mathcal{G}_{s_k}$ only contains items in domain k, namely, the items that user u has interacted with in this domain. Therefore, the extracted user embedding $\hat{e}_u^{s_k}$ in $\mathcal{G}_{s_k}$ reflects u's domain-specific interest. To combine the two viewpoints, we need to align the user embeddings extracted from domain-shared and domain-specific graphs, making them as close as possible. Specifically, we employ parallel GNNs respectively for the user embedding extraction and minimize the distance between the two representations, measured by cosine similarity:

$$\mathcal{D}_{\cos}(c, s_k) = \sum_{u\in\mathcal{U}} (1 - \cos(\hat{e}_u^c, \hat{e}_u^{s_k})). \tag{12}$$

4.4.2. Domain-specific interest regularization

The above domain-shared strategy enforces the user interests in domain-shared graph $\mathcal{G}_c$ and those in domain-specific graphs $\{\mathcal{G}_{s_1}, \mathcal{G}_{s_2}, \dots, \mathcal{G}_{s_K}\}$ to be as close to each other as possible, so as to transfer knowledge across different domains. However, this may lead to over-smoothing of user interests across domains, meaning that the uniqueness of each domain is no longer obvious. Therefore, we introduce another regularization strategy, which is able to distinguish domain-specific interests from one domain to another. Following our strategy, the domain-specific interests retain as much domain-specific information as possible. Essentially, we encourage domain-specific features to encode user's interests from different aspects. To achieve this goal, we incorporate the orthogonal constraint as part of our loss, as shown in the following Equation (13). Many previous studies [44,45] have also demonstrated its effectiveness.

$$\mathcal{D}_{\mathrm{orth}}(s_k, s_j) = \sum_{u\in\mathcal{U}} |\hat{e}_u^{s_k\,\mathrm{T}} \hat{e}_u^{s_j}|, \tag{13}$$

where $|\cdot|$ denotes L_1-norm. $\hat{e}_u^{s_j}$ represents the refined user representation of user u extracted from the j-th domain.

4.4.3. Shared-specific interest gated ensemble

We can represent the learned user interest as a sphere, where the shared component corresponds to the sphere's center, which is similar across different domains, while the specific component diverges from the sphere in different directions. In this way, the two user interest components can be integrated into a complete representation. We propose a novel gated neural component g_u, which can adaptively control feature intersection between domain-shared and domain-specific user interests.

$$\hat{y}_{uv}^m = g_u \cdot \hat{y}_{uv}^c + (1-g_u)\cdot \hat{y}_{uv}^{s_k}, \tag{14}$$

where g_u represented the gate component. Specifically, we have:

$$g_u = w_{g_2}^{\mathrm{T}} \mathrm{ReLU}\big((\hat{e}_u^c \parallel \hat{e}_u^{s_k}) W_{g_1} + b_{g_1}\big), \tag{15}$$

where $\hat{y}_{uv}^c$, $\hat{y}_{uv}^{s_k}$ and $\hat{y}_{uv}^m$ represent the domain-shared user interest of user u, the domain-specific user interest of user u on domain k, and the ensembled user interests of user u, respectively. $W_{g_1} \in \mathbb{R}^{Ld\times d}, b_{g_1} \in \mathbb{R}^d, w_{g_2} \in \mathbb{R}^d$ are gate component's parameters. Moreover,

$\hat{y}_{uv}^{m}$ can be treated as an online teacher and the ensemble user interests can be distilled back into two graphs to optimize student interest $\hat{y}_{uv}^{c}$ and $\hat{y}_{uv}^{s_k}$ in a closed form. The alignment of user interests between a teacher and a student can be defined as:

$$\mathcal{D}_{\text{ce}}(c, s_k, m) = \sum_{(u,v)\in\mathcal{Y}\cup\mathcal{Y}^-} \mathcal{L}_{\text{ce}}\left(\sigma(\frac{\hat{y}_{uv}^{c}}{\tau}), \sigma(\frac{\hat{y}_{uv}^{m}}{\tau})\right) + \mathcal{L}_{\text{ce}}\left(\sigma(\frac{\hat{y}_{uv}^{s_k}}{\tau}), \sigma(\frac{\hat{y}_{uv}^{m}}{\tau})\right), \tag{16}$$

where σ denotes the logistic function, τ serves as a temperature parameter, making the label probability distribution softer. Compared with doing exact matching on the score labels, ensemble soft labels provide a wealth of information about diverse user interests, which helps optimize student models more effectively. In this way, the user's common interest shared across domains and specific interest in the corresponding domain can be integrated and mutually learned from the ensemble interest.

4.5. Model optimization

We combine the mutual regularization loss to our multi-domain recommendation loss, and the objective for the given domain-shared graph $\mathcal{G}_c$ and specific-domain graph $\mathcal{G}_{s_k}$ can be respectively defined as:

$$\begin{aligned} \mathcal{L}_c &= \mathcal{L}_{\Theta_c} + \sum_{j=1}^{K} \mathcal{D}_{\cos}(c, s_j), \\ \mathcal{L}_{s_k} &= \mathcal{L}_{\Theta_{s_k}} + \sum_{j=1, j\neq k}^{K} \mathcal{D}_{orth}(s_k, s_j). \end{aligned} \tag{17}$$

The overall objective of our mutual regularization learning framework is:

$$\mathcal{L} = \mathcal{L}_c + \mathcal{L}_{s_k} + \mathcal{L}_{\Theta_m} + \tau^2 * \mathcal{D}_{\text{ce}}(c, s_k, m), \tag{18}$$

where τ^2 is the weight to help us ensure that the relative contributions of the ground-truth label (i.e., hard label) and teacher labels' (i.e., soft label) probability distributions remain the same.

We utilize mini-batch training to compute the gradient and mutual regularization strategies are performed during each update step throughout the training phase. For every epoch of training, we compute the target user-item pair matching score $\hat{y}_{uv}^{c}$, $\hat{y}_{uv}^{s_k}$ and $\hat{y}_{uv}^{m}$, where k is decided by the domains this pair belongs to. Then we compute the model objective in both the domain-shared graph and the domain-specific ones, and update the parameters according to the predictions and mutual regularization strategies. The overall optimization steps are shown in detail in Algorithm 1.

Algorithm 1: Graph Mutual Regularization Learning.

Input: Training user-item interactions set $\mathcal{Y}\cup\mathcal{Y}^-$, domain-shared graph $\mathcal{G}_c$, domain-specific graphs $\{\mathcal{G}_{s_1} \dots \mathcal{G}_{s_K}\}$
Initialize: Models Θ_c and $\Theta_{s_1} \dots \Theta_{s_K}$ with different initialization.
Repeat:
1 Randomly sample user-item pair u, v from $\mathcal{Y}\cup\mathcal{Y}^-$;
2 Get the domain index k of user-item pair u, v;
3 Update $\hat{e}_u^c, \hat{e}_v^c, \hat{e}_u^{s_k}, \hat{e}_v^{s_k}, \hat{y}_{uv}^c, \hat{y}_{uv}^{s_k}$ in $\mathcal{G}_c$ and $\mathcal{G}_{s_k}$; // Eq. (8)-(9)
4 Compute the $\mathcal{D}_{\cos}(c, s_k), \mathcal{D}_{\text{orth}}(s_k, s_j)$; // Eq. (12)-(13)
5 Compute the ensemble user interest $\hat{y}_{uv}^m$; // Eq. (14)
6 Distil the ensemble interest back ; // Eq. (16)
7 Compute the final overall loss and update ; // Eq. (18)
Until *convergence*;

4.6. Complexity analysis

For the multi-domain recommendation framework defined in previous sections, the computational consumption is mainly composed of two parts: (i) the domain-shared parallel GNN module; (ii) the domain-specific parallel GNN module; (iii) the mutual regularization strategy. Assume the batch size is B, given the constructed domain-shared graph with an average of M_B users and N_B items, and the constructed domain-specific graph with an average of $M_{B,k}$ users and $N_{B,k}$ items, the number of domain and GNN layer are K and L respectively, and the representation dimension is d. For (i) and (ii), the time complexity of the corresponding parallel GNN module is $O(B(M_B + N_B)Ld)$ and $O(BK(M_{B,k} + N_{B,k})Ld))$, respectively. For (iii), the time complexity of domain-shared and domain-specific regularization strategy is $O(BKd)$ and $O(BK^2d)$. To sum up, we have the overall time complexity of GMR-Rec, $O(B((M_B + KM_{B,k} + N_B + KN_{B,k})L + K^2 + K)d)$, which scales linearly w.r.t. the number of users and items in the constructed graph. Moreover, GMR-Rec requires one domain-shared graph with K domain-specific graph, and the space complexity is $O(B(M_B + KM_{B,k} + N_B + KN_{B,k}) + Kd)Ld)$.

5. Experiment

We use four real-world data sets to verify our model's performance compared to the other baselines. Our experiments are aiming at answering the three important questions listed below:

Table 1
Descriptive statistics of four datasets.

Dataset	#Domains	#Users	#Items	#Interactions
Dianping	#POI	18,636	11,372	133,016
	#Feeds	10,631	24,460	256,244
	Total	21,737	35,832	389,260
Movie	#Netflix	35,072	1,536	647,096
	#MovieLens	7,402	3,837	513,304
	Total	42,474	4,696	1160,400
Douban	#Book	6,707	3,073	52,453
	#Movie	14,870	9,788	616,164
	#Music	2,755	1,610	16,981
	Total	14,956	14,471	685,598
Amazon	#Cell	21,182	4,846	69,798
	#Electronic	27,915	17,962	337,511
	#Software	1,238	303	2,787
	#CD	17,640	27,494	394,765
	Total	45,136	50,605	804,861

RQ1: How well does GMR-Rec perform on multi-domain recommendations compared to the other state-of-the-art models? Can GMR-Rec successfully mitigate the cold-start problem and the data sparsity problems by transferring knowledge from other domains to the target domain?

RQ2: How does GMR-Rec benefit from its key components, such as mutual regularization learning and how do different hyper-parameters in key components (e.g., neighborhood hops, self-attention layer numbers) affect the results of GMR-Rec?

RQ3: Can GMR-Rec provide an interpretable analysis of mutual regularization learning w.r.t. the domain-shared interest and the domain-specific interests of the user, as well as further providing such interpretable analysis to the distilled knowledge learned by each model?

5.1. Experimental settings

5.1.1. Data sets

We selected four real-world multi-domain recommendation datasets collected from online platforms. Table 1 summarizes the statistics of datasets.

- **Dianping**[1] was gathered from Meituan Dianping, a popular social media platform that contains information and reviews of restaurants, hotels, entertainment, movies, etc. We choose one-month interaction records of a city in both POI and Feeds domains.
- **Movie**[2] is a movie-shared rating dataset with common movies and different users in both Netflix and Movielens datasets. We merge the same movies from the Netflix and Movielens datasets, Then, we learn the domain-shared representation and the domain-specific representations for each of the items instead.
- **Douban**[3] is a community website where users can describe and share reviews. We utilize the data provided in the previous work [46] and select three related domains, namely, Book, Music, and Movie.
- **Amazon**[4] is a product recommendation dataset that records user ratings and reviews of products from various categories [47]. We select four categories, namely Cell Phones (Cell), Electronics (Elec), Software and CD, as four domains.

For Amazon, Douban and Movie, we use the core set to filter interactions. We chronologically rank the interactions of each dataset and select the top 80% of historical interactions to constitute the training set. The remaining 20% are split into half-and-half, serving as the validation and test sets.

5.1.2. Evaluation metrics

Since ranking all items for each user is time-consuming, we use the leave-one-out approach [48] for evaluation. For each positive sample of the user-item interactions observed, we uniformly at random sample 99 items from those that the user has never interacted with as negative samples, and then we rank the test item among all 100 items. To assess the ranked list with the ground-truth item set (GT), we employ two widely used metrics for all models to evaluate the performance: Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG). We truncate the ranking list to K for both metrics, denoted as $HR@K$ and $NDCG@K$, calculated as:

[1] https://www.dianping.com.
[2] https://www.kaggle.com/datasets/netflix-inc/netflix-prize-data; https://www.kaggle.com/datasets/grouplens/movielens-20m-dataset.
[3] https://www.douban.com.
[4] http://jmcauley.ucsd.edu/data/amazon/.

$$HR@K = \frac{Number\ of\ Hits@K}{|GT|},$$
$$NDCG@K = \frac{1}{maxDCG} \sum_{i=1}^{K} \frac{2^{r_i} - 1}{\log_2(i+1)}, \tag{19}$$

where a hit is recorded if a test item appears in the recommended list, r_i is the graded relevance of the item at position i, and $maxDCG$ is the normalization factor. We set $K = 10$ and use the simple binary relevance following previous work [48,20], namely $r_i = 1$ if the item is in the test set, and 0 otherwise.

5.1.3. Baselines

To evaluate the model effectiveness, we compare GMR-Rec with three classes of models: (A) Classical CF models, (B) GNN-based recommendation models and (C) Multi-domain recommendation models.

- **BPR-MF** [49] (A): This is the matrix factorization-based classical CF method. The method employs the Bayesian Personalized Ranking (BPR) as the loss.
- **NMF** [48] (A): This method is also matrix factorization-based. The method uses Multilayer Perceptron (MLP) to model the nonlinear feature interactions of users and items.
- **PinSAGE** [25] (B): This method aims to apply GraphSAGE [22] on the item-item graph, and we employ the method on user-item interaction graphs for comparison.
- **NGCF** [6] (B): This method obtains high-quality user and item embeddings via multiple embedding propagation layers to explicitly model the higher-order connectivity between user and item.
- **LightGCN** [7] (B): The method further simplifies the NGCF's feature propagation component by removing the non-linear activation function and the transformation matrices.
- **CMF** [3] (C): This method simultaneously factorizes matrices of each domain, sharing factors of common users in multiple domains for recommendation.
- **CDFM** [15] (C): This method incorporates information of auxiliary domain w.r.t. shared users for the target domain as context and employs Factorization Machine (FM) for recommendation.
- **DDTCDR** [5] (C): This is a cross-domain recommendation approach that utilizes dual transfer learning across different domains and applies a latent orthogonal mapping to preserve the similarity of a user's interests across these domains. We further extend this model to support multi-domains.
- **UniCDR** [20](C): This method provides a unified framework for multi-domain recommendation, which learns both domain-shared and domain-specific user representations respectively and models the correlation of two representations via masking mechanism and contrastive loss.

To more accurately evaluate the performance, for classical CF and GNNs-based methods, we use the same methods but with the multi-domain mixed dataset as their training data set. By this way, we keep all the baselines using the same training set as our model, and that makes a fair comparison.

5.1.4. Implementation detail

Our GMR-Rec model is implemented by Pytorch. For all models, The embedding size is fixed to 100. For our model, we sample 2 hops of neighborhoods with the number of neighbors being 10 and 50 respectively, and the number of cross-hops self-attention layers is set to 2 in the default setting. We set the temperature parameter to 1 for mutual regularization. GMR-Rec is trained with Adam optimizer via early stopping with the patience of 20, i.e., the training process is stopped if NDCG@10 on validation set does not keep increase in 20 consecutive epochs. For all baselines, we employ a gird search carefully to find the best hyper-parameters.

5.2. Performance comparison (RQ1)

5.2.1. Overall comparison

The test results of models on four real-world datasets are summarized in Table 2 and we have the following findings:

- GNN-based recommendation models, which leverage information propagation for high-order connectivity modeling in a user-item bipartite graph, outperform the classical CF method especially BPRMF in most cases. However, when it comes to NMF and multi-domain methods, the performance of GNN-based methods is sometimes worse, demonstrating that common user interest explored by explicit knowledge transfer in the constructed domain-shared graph is insufficient.
- Multi-domain recommendation methods, which transfer knowledge from other domains to help with the recommendation in the target domain, achieve better performance than most of the classic CF methods and most of the GNNs-based methods. This indicates that the transfer learning mechanism for multi-domain recommendation is more effective compared with GNN-based methods, which are not specifically designed for knowledge transfer across domains. However, the improvement brought by these methods is marginal, and the domain-specific interest of a user in each domain is not yet considered.
- GMR-Rec consistently outperforms other baselines on all of the four real-world datasets. GMR-Rec is specifically designed for multi-domain recommendation, and therefore can model both the domain-shared interest and the domain-specific interests of

Table 2
Performance results on four different datasets. Here, ** and * denotes the statistical significance for $p <= 0.01$ and $p <= 0.05$ compared to the best baseline model.

Dataset	Domain	Metrics	Classical		GNNs-based			Multi-domain				Ours
			BPRMF	NMF	PinSAGE	NGCF	LightGCN	CMF	CDFM	DDTCDR	UniCDR	GMR-Rec
Dianping	POI	HR@10	0.5516	0.6702	0.6812	0.6716	0.6936	0.7126	0.7067	0.7035	0.7122	**0.7139**
		NDCG@10	0.3765	0.4459	0.4524	0.4472	0.4662	0.4845	0.4820	0.4758	0.4875	**0.4996****
	Feeds	HR@10	0.5160	0.5502	0.5813	0.5702	0.5892	0.6205	0.5924	0.6023	0.6386	**0.6533****
		NDCG@10	0.3289	0.3385	0.3819	0.3731	0.3830	0.4026	0.3896	0.3868	0.4157	**0.4307****
Movie	Netflix	HR@10	0.4305	0.6295	0.6130	0.5859	0.5904	0.6810	0.6688	0.6589	0.6989	**0.7123***
		NDCG@10	0.2485	0.3778	0.3646	0.3350	0.3592	0.4604	0.4403	0.4402	0.4623	**0.4799****
	MovieLens	HR@10	0.2432	0.5027	0.4990	0.4628	0.4860	0.5869	0.5529	0.5616	0.5962	**0.6193****
		NDCG@10	0.1307	0.2926	0.2882	0.2533	0.2649	0.3708	0.3667	0.3545	0.3860	**0.4040****
Douban	Book	HR@10	0.3199	0.4077	0.3768	0.3288	0.4034	0.4231	0.4224	0.3941	0.4216	**0.4262***
		NDCG@10	0.2073	0.2518	0.2270	0.1943	0.2494	0.2613	0.2612	0.2443	0.2615	**0.2643***
	Movie	HR@10	0.4050	0.4568	0.4742	0.4714	0.4853	0.4819	0.4695	0.4566	0.4806	**0.5093****
		NDCG@10	0.2485	0.2707	0.2841	0.2816	0.2870	0.2853	0.2806	0.2731	0.2813	**0.3011****
	Music	HR@10	0.1385	0.1866	0.2129	0.1990	0.1889	0.1913	0.1618	0.1791	0.2007	**0.2198***
		NDCG@10	0.0730	0.1034	0.1097	0.1131	0.1033	0.1026	0.0903	0.0956	0.1042	**0.1179***
Amazon	Cell	HR@10	0.3307	0.3746	0.3635	0.3586	0.3878	0.3904	0.3920	0.3842	0.3936	**0.3998****
		NDCG@10	0.1992	0.2285	0.2185	0.2040	0.2332	0.2358	0.2359	0.2307	0.2365	**0.2452****
	Electronic	HR@10	0.3467	0.3521	0.3967	0.3647	0.3947	0.4160	0.4138	0.4073	0.4165	**0.4276****
		NDCG@10	0.2076	0.2141	0.2369	0.2176	0.2365	0.2481	0.2474	0.2392	0.2487	**0.2589****
	Software	HR@10	0.2341	0.2878	0.2926	0.2961	0.2996	0.3073	0.3021	0.2998	0.3071	**0.3173****
		NDCG@10	0.1892	0.2154	0.2174	0.2213	0.2224	0.2321	0.2314	0.2267	0.2342	**0.2447****
	CD	HR@10	0.2424	0.2573	0.4228	0.4258	0.4276	0.3357	0.3946	0.4216	0.4312	**0.4535****
		NDCG@10	0.1319	0.1444	0.2363	0.2399	0.2408	0.1838	0.1942	0.2403	0.2507	**0.2635****

a particular user, through the mutual learning of our proposed framework using parallel GNNs. The significant improvements indicate a positive impact on achieving better representations for the multi-domain recommendation.

5.2.2. *Cold-start recommendation*

The multi-domain recommendation is especially useful for alleviating the cold start and sparsity issue by transferring knowledge from other domains. We investigate whether GMR-Rec can better alleviate this issue by comparing the performance with other multi-domain recommendation methods on different sparse distributions of the data. To this end, we train our model and the baselines with 20%, 40%, 60%, and 80% of the entire training data. The results are illustrated in Fig. 4 w.r.t. different sparse distributions of training data on Dianping. The Amazon and Douban datasets exhibit similar performance and are not discussed in detail due to the space limit. It can be seen that the GMR-Rec consistently outperforms other multi-domain recommendation baselines over all sparse distribution of the training data, and the improvement is particularly significant in the Feeds domain. The corresponding results show that our proposed GMR-Rec can effectively improve the multi-domain recommendation performance to alleviate the cold start and data sparsity issue.

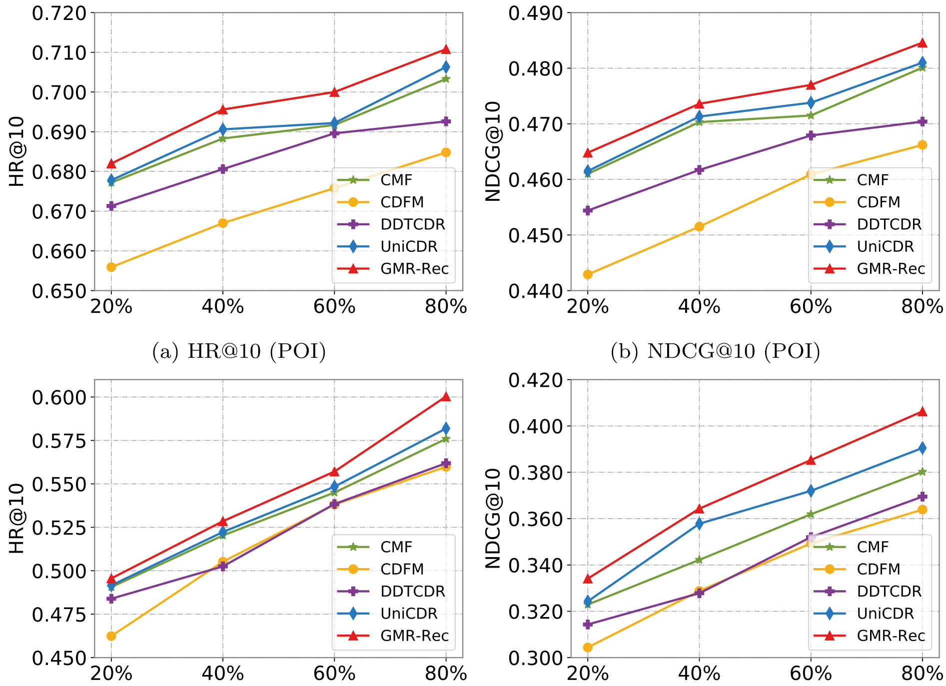
5.3. *Study of GMR-Rec (RQ2)*

As parallel GNNs and their mutual regularization learning play pivotal roles in GMR-Rec, we investigate their impact on performance. We first conduct mutual regularization learning on two other baselines, NMF [48] and LightGCN [7]. Then ablation studies are performed to explore how different learning strategies affect the performance. Moreover, we study how the number of hops and self-attention layers in parallel GNN affects the performance.

5.3.1. *Effect of mutual regularization learning*

To evaluate the effect of mutual regularization learning, we incorporate baseline models NMF and LightGCN with mutual regularization learning and perform ablation studies to show how mutual regularization learning affects the model performance. Specifically, we compare different recommendation performance under three conditions, namely user-item interaction data in one domain (Specific), interaction data in all domains (Shared) and interaction data in all domains with our proposed mutual regularization learning (Mutual). Fig. 5 summarizes the results and we have the following findings:

- Comparing our proposed parallel GNN to LightGCN and NMF, we find that our model consistently outperforms other two baselines under three conditions. This could justify the effectiveness of the proposed parallel GNN framework, especially on the data with different domains.



(a) HR@10 (POI) (b) NDCG@10 (POI)

(c) HR@10 (Feeds) (d) NDCG@10 (Feeds)

Fig. 4. Performance results over the sparse distributions of data on *Dianping*.

- Focusing on LightGCN and NMF, we find that learning to get domain-shared interest outperforms learning to get domain-specific interest. Meanwhile, mutual regularization in domain-shared and domain-specific graphs leads to higher performance which illustrates its effectiveness for the multi-domain recommendation.
- Focusing on GMR-Rec, we find that its performance improves less when learning in the domain-shared graph. This indicates that common user interest across different domains is insufficient, thus it is necessary to consider the domain-specific interest. And we can see further improvement by incorporating mutual regularization learning strategies.

5.3.2. Effect of neighborhood Hops

GMR-Rec aggregates neighborhood information from various hops independently. To investigate how GMR-Rec is affected by these multiple hops of neighborhoods, we vary the depth of the model. In particular, We fix other hyperparameters and select the maximum number of hops from $\{1, 2, 3\}$ to verify its impact on the performance. The experimental results are shown in Table 3. From these results, we have the following findings:

- Incorporating multiple hops allows the target node to gather collaborative signals carried by high-order connections and increase the depth of the model. 2 hops of neighborhoods in the model consistently outperform 1 hop across all the datasets, since 1 hop of neighborhoods only contains items (users) that the target node has interacted with.
- When further aggregating neighborhood information beyond 2 hops away, the impact is negligible and may lead to overfitting. This could be attributed to the use of deep architectures and the incorporation of noise, which occurs when distant, unrelated neighborhood information is included in the representation learning process.

5.3.3. Effect of self-attention layer numbers

To investigate how embedding propagation affects the performance, we fix the maximum hops of neighbors in parallel GNN to 2 and consider the model with the varying number of self-attention layers in the cross-hops neighborhood propagation. As shown in Fig. 6, we set the number of self-attention layers in the range of $\{0, 1, 2, 3, 4\}$ and have the following findings:

- When there is no embedding propagation between different hops of neighbors for the target nodes (number of self-attention layers equals 0), our model only concatenates neighborhood information independently aggregated from each hop. The performance of the model becomes worse, suggesting that embedding propagation in the parallel GNN framework can greatly boost recommendation performance.
- Increasing the number of self-attention layers can substantially enhance the recommendation performance. When the number of layers reaches 3, the performance becomes stable, showing that embedding propagation many times might cause the neighborhood information over-smoothing and less informative for the recommendation.

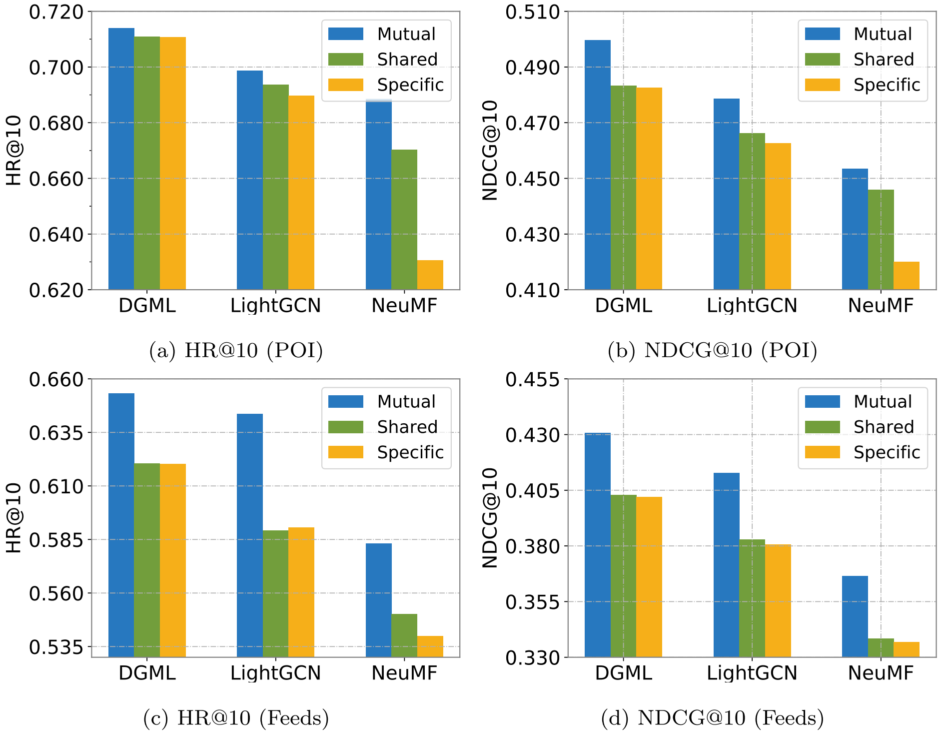


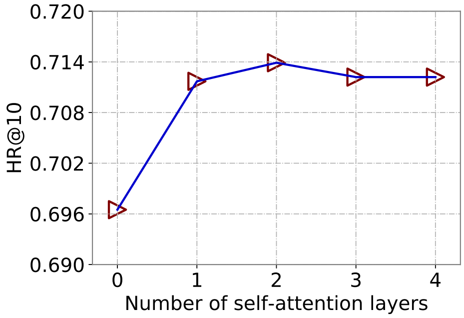
Fig. 5. Effect of the mutual regularization learning strategy on *Dianping*.

Table 3
Performance results w.r.t. different number of neighborhood hops.

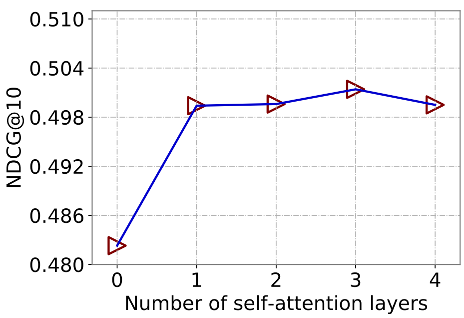
Dataset	Hops	Domains	HR@10	NDCG@10
Dianping	1	POI	0.7096	0.4913
		Feeds	0.6457	0.4262
	2	POI	0.7139	0.4996
		Feeds	0.6533	0.4307
	3	POI	0.7049	0.4901
		Feeds	0.6462	0.4245
Movie	1	Netflix	0.6942	0.4613
		MovieLens	0.5942	0.3851
	2	Netflix	0.7123	0.4799
		MovieLens	0.6193	0.4040
	3	Netflix	0.7064	0.4756
		MovieLens	0.6035	0.3947
Douban	1	Book	0.4139	0.2568
		Movie	0.4953	0.2964
		Music	0.2165	0.1157
	2	Book	0.4262	0.2643
		Movie	0.5093	0.3011
		Music	0.2198	0.1179
	3	Book	0.4221	0.2622
		Movie	0.5113	0.3028
		Music	0.2230	0.1167

5.4. Case study and visualization (RQ3)

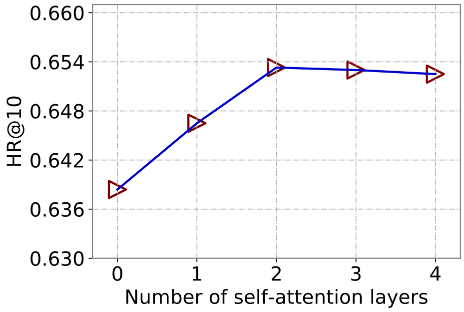
To illustrate how GMR-Rec facilitates the multi-domain recommendation task and corresponding embedding learning. We explore the attention weight in cross-hops neighborhood propagation and visualize the node embeddings of GMR-Rec for qualitative evaluation of the embedding results.



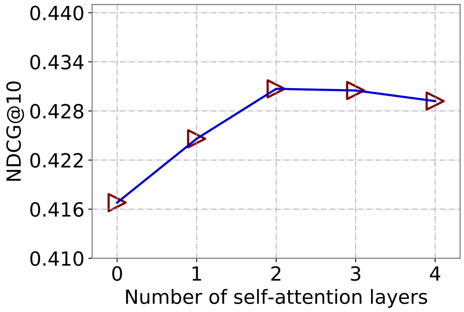
(a) HR@10 (POI)



(b) NDCG@10 (POI)



(c) HR@10 (Feeds)



(d) NDCG@10 (Feeds)

Fig. 6. Performance results w.r.t. different number of self-attention layers on *Dianping*.

5.4.1. Exploring attention weight

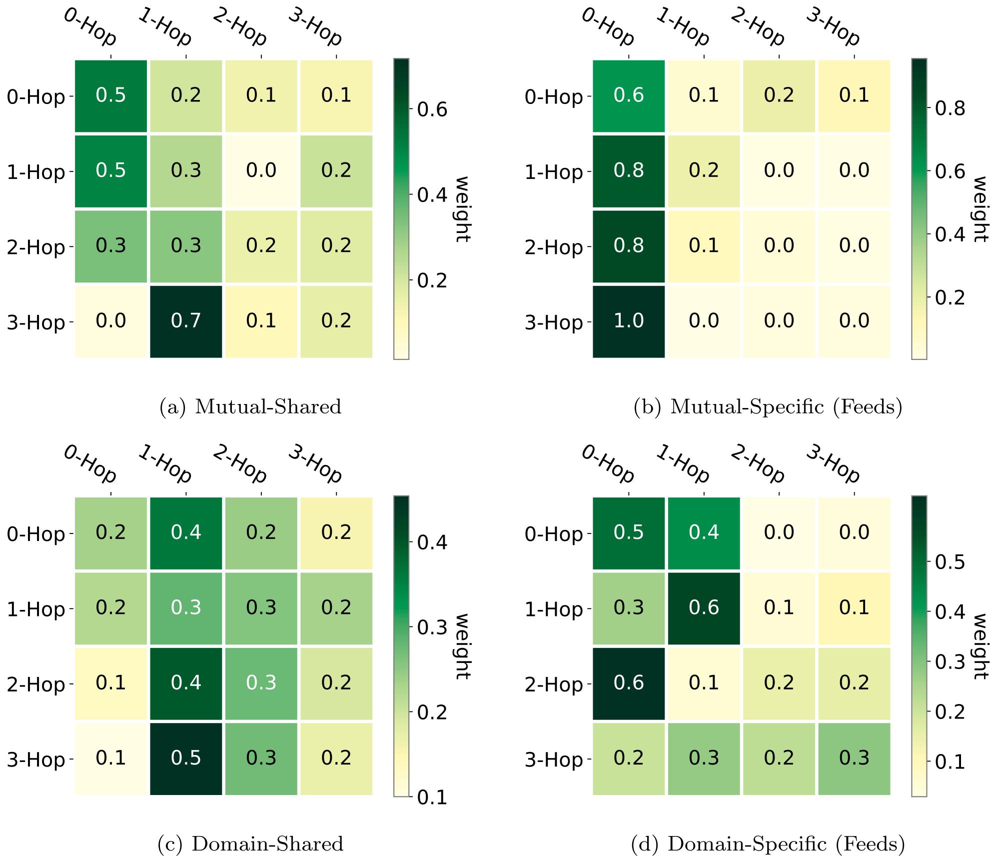
We select users as target nodes and plot the average attention heat maps of parallel GNN across different hops in both domain-shared and domain-specific (Feeds domain) graphs with mutual regularization strategies. As shown in Fig. 7b, we can find that the attention of different hops of neighbors is more focused on the target node (0-Hop), which demonstrates the specific feature of the target node is more important in the graph of the Feeds domain. In contrast, as shown in Fig. 7a, feature interactions between 2-hops of neighbors or even faraway neighbors of the target node still have a large attention weight in the domain-shared graph. This is very reasonable since users in the domain-shared graph prefer to transfer knowledge across domains. Moreover, we also visualize the attention weight of domain-shared and domain-specific graphs without mutual regularization strategies. As shown in Fig. 7c and Fig. 7d, we can find that 1-Hop neighbors have higher attention weights in both graphs, indicating different patterns compared to scenarios with mutual regularization strategies.

5.4.2. Visualization

For a more intuitive comparison between common and specific user interests, we randomly select six users and their relevant items in different domains from Dianping datasets and visualize the learned embeddings. The results derived from domain-shared and domain-specific graphs with mutual regularization strategies are shown in Fig. 8a and Fig. 8b. We can easily tell that learned embeddings mix together in the domain-shared graph while separating into two groups according to their domains in domain-specific graphs. This highlights the distinct contributions of domain-shared and domain-specific user interests, demonstrating the necessity of integrating them together within our model. In contrast, domain-shared and domain-specific user and item embeddings without mutual regularization strategies are shown in Fig. 8c and Fig. 8d. We can find that the learned user and item embeddings of both graphs are mixed together in the nearby regions of the space, failing to capture the distinct characteristics of the two graphs.

6. Conclusion and future work

In this work, we present a multi-domain recommendation model, GMR-Rec. To model both the domain-shared common interest and the domain-specific user interests across the different domains, we separately build domain-shared and domain-specific user-item bipartite graphs and extract corresponding user features via proposed parallel GNN models. Moreover, we apply different mutual regularization learning strategies to encourage domain-specific features can be distinguished from other features while keeping these different features close to the domain-shared feature. Based on that, domain-shared and domain-specific user interests can complete knowledge with each other and mutually learn to build the complete user interest for the recommendation. Extensive experiments that are conducted on four real-world datasets further demonstrate the superiority of our model. However, the temporal influence on both domain-shared and domain-specific user interests is overlooked.



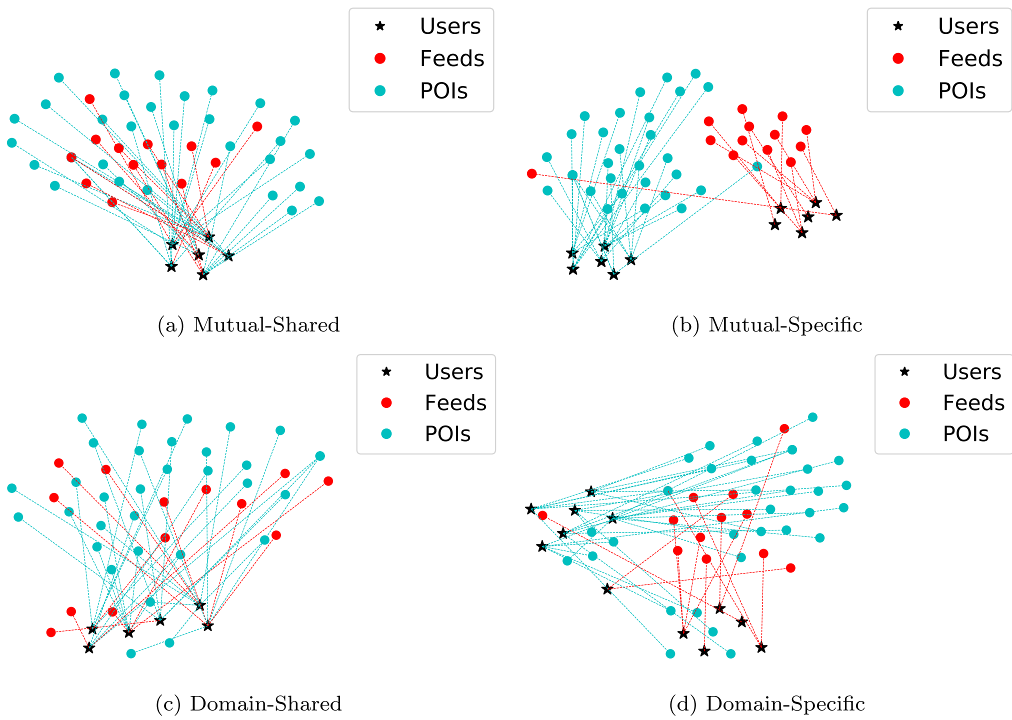
(a) Mutual-Shared

(b) Mutual-Specific (Feeds)

(c) Domain-Shared

(d) Domain-Specific (Feeds)

Fig. 7. Heat maps of attention weights on *Dianping*. The first and second rows correspond to weights w/ and w/o mutual regulation learning strategy, respectively.



(a) Mutual-Shared

(b) Mutual-Specific

(c) Domain-Shared

(d) Domain-Specific

Fig. 8. Visualization of user and item representations learned on t-SNE. Here stars denote users from *Dianping* dataset, and points represent items in the same domain.

In future work, we will explore whether GMR-Rec can extract both short-term and long-term user interests in temporal interactions. Furthermore, we also plan to explore the potential of GMR-Rec for general graph machine-learning tasks, such as unsupervised and semi-supervised node classification.

CRediT authorship contribution statement

Yifan Wang: Writing – review & editing, Writing – original draft, Methodology, Investigation, Formal analysis, Conceptualization. **Yangzi Yang:** Writing – review & editing, Investigation, Conceptualization. **Shuai Li:** Writing – review & editing, Methodology, Data curation. **Yutao Xie:** Validation, Methodology. **Zhiping Xiao:** Writing – review & editing, Writing – original draft, Methodology, Investigation. **Ming Zhang:** Supervision, Project administration, Funding acquisition. **Wei Ju:** Supervision, Investigation, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The paper is supported by "the Fundamental Research Funds for the Central Universities" in UIBE (Grant No. 23QN02) and the National Natural Science Foundation of China (NSFC Grant Numbers 62306014 and 62276002).

Data availability

Data will be made available on request.

References

[1] Y. Koren, R.M. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, IEEE Computer 42 (8) (2009) 30–37.
[2] B. Li, Q. Yang, X. Xue, Can movies and books collaborate? Cross-domain collaborative filtering for sparsity reduction, in: IJCAI, 2009.
[3] A.P. Singh, G.J. Gordon, Relational learning via collective matrix factorization, in: SIGKDD, 2008, pp. 650–658.
[4] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, C. Zhu, Personalized recommendation via cross-domain triadic factorization, in: WWW, 2013, pp. 595–606.
[5] P. Li, A. Tuzhilin, Ddtcdr: deep dual transfer cross domain recommendation, in: WSDM, 2020, pp. 331–339.
[6] X. Wang, X. He, M. Wang, F. Feng, T.-S. Chua, Neural graph collaborative filtering, in: SIGIR, 2019, pp. 165–174.
[7] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, M. Wang, Lightgcn: simplifying and powering graph convolution network for recommendation, in: SIGIR, 2020, pp. 639–648.
[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: NeurIPS, 2017, pp. 5998–6008.
[9] Y. Wang, Y. Qin, Y. Han, M. Yin, J. Zhou, H. Yang, M. Zhang, Ad-aug: adversarial data augmentation for counterfactual recommendation, in: ECML-PKDD, 2022, pp. 474–490.
[10] W. Ju, S. Yi, Y. Wang, Q. Long, J. Luo, Z. Xiao, M. Zhang, A survey of data-efficient graph learning, in: IJCAI, 2024, pp. 8104–8113.
[11] Y. Zhang, B. Cao, D.-Y. Yeung, Multi-domain collaborative filtering, in: UAI, 2010, pp. 725–732.
[12] Z. Zhang, X. Jin, L. Li, G. Ding, Q. Yang, Multi-domain active learning for recommendation, in: AAAI, 2016.
[13] X. Hao, Y. Liu, R. Xie, K. Ge, L. Tang, X. Zhang, L. Lin, Adversarial feature translation for multi-domain recommendation, in: SIGKDD, 2021, pp. 2964–2973.
[14] H. Li, Y. Wang, Z. Xiao, J. Yang, C. Zhou, M. Zhang, W. Ju, et al., DisCo: graph-based disentangled contrastive learning for cold-start cross-domain recommendation, arXiv preprint, arXiv:2412.15005, 2024.
[15] B. Loni, Y. Shi, M. Larson, A. Hanjalic, Cross-domain collaborative filtering with factorization machines, in: ECIR, 2014, pp. 656–661.
[16] B. Jin, X. Xu, Wholesale price forecasts of green grams using the neural network, Asian J. Econ. Bank. (2024).
[17] B. Jin, X. Xu, Price forecasting through neural networks for crude oil, heating oil, and natural gas, Measurement: Energy 1 (1) (2024) 100001.
[18] T. Man, H. Shen, X. Jin, X. Cheng, Cross-domain recommendation: an embedding and mapping approach, in: IJCAI, 2017, pp. 2464–2470.
[19] G. Hu, Y. Zhang, Q. Yang, Conet: collaborative cross networks for cross-domain recommendation, in: CIKM, 2018, pp. 667–676.
[20] J. Cao, S. Li, B. Yu, X. Guo, T. Liu, B. Wang, Towards universal cross-domain recommendation, in: WSDM, 2023, pp. 78–86.
[21] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: ICLR, 2017.
[22] W.L. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: NeurIPS, 2017, pp. 1024–1034.
[23] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: ICLR, 2018.
[24] Y. Wang, X. Luo, C. Chen, X.-S. Hua, M. Zhang, W. Ju, DisenSemi: semi-supervised graph classification via disentangled representation learning, IEEE Trans. Neural Netw. Learn. Syst. (2024).
[25] R. Ying, R. He, K. Chen, P. Eksombatchai, W.L. Hamilton, J. Leskovec, Graph convolutional neural networks for web-scale recommender systems, in: SIGKDD, 2018, pp. 974–983.
[26] S. Fan, J. Zhu, X. Han, C. Shi, L. Hu, B. Ma, Y. Li, Metapath-guided heterogeneous graph neural network for intent recommendation, in: SIGKDD, 2019, pp. 2478–2486.
[27] Y. Wang, S. Tang, Y. Lei, W. Song, S. Wang, M. Zhang, Disenhan: disentangled heterogeneous graph attention network for recommendation, in: CIKM, 2020, pp. 1605–1614.
[28] C. Zhao, C. Li, C. Fu, Cross-domain recommendation via preference propagation graphnet, in: CIKM, 2019, pp. 2165–2168.
[29] M. Liu, J. Li, G. Li, P. Pan, Cross domain recommendation via bi-directional transfer graph collaborative filtering networks, in: CIKM, 2020, pp. 885–894.
[30] Q. Cui, T. Wei, Y. Zhang, Q. Zhang, Herograph: a heterogeneous graph framework for multi-target cross-domain recommendation, in: ORSUM@ RecSys, 2020.
[31] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, arXiv preprint, arXiv:1503.02531, 2015.
[32] A. Romero, N. Ballas, S.E. Kahou, A. Chassang, C. Gatta, Y. Bengio, Fitnets: hints for thin deep nets, arXiv preprint, arXiv:1412.6550, 2014.
[33] T. Furlanello, Z. Lipton, M. Tschannen, L. Itti, A. Anandkumar, Born again neural networks, in: ICML, 2018, pp. 1607–1616.

[34] G. Chen, W. Choi, X. Yu, T. Han, M. Chandraker, Learning efficient object detection models with knowledge distillation, in: NeurIPS, 2017, pp. 742–751.
[35] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, Q. Liu, Tinybert: distilling bert for natural language understanding, arXiv preprint, arXiv:1909.10351, 2019.
[36] Z. Xiao, X. Xu, H. Xing, B. Zhao, X. Wang, F. Song, R. Qu, L. Feng, Dtcm: deep transformer capsule mutual distillation for multivariate time series classification, IEEE Trans. Cogn. Dev. Syst. (2024).
[37] Z. Xiao, H. Tong, R. Qu, H. Xing, S. Luo, Z. Zhu, F. Song, L. Feng, Capmatch: semi-supervised contrastive transformer capsule with feature-based knowledge distillation for human activity recognition, IEEE Trans. Neural Netw. Learn. Syst. (2023).
[38] H. Zhang, S. Lin, W. Liu, P. Zhou, J. Tang, X. Liang, E.P. Xing, Iterative graph self-distillation, in: ICLR, 2020.
[39] Y. Zhang, X. Xu, H. Zhou, Y. Zhang, Distilling structured knowledge into embeddings for explainable and accurate recommendation, in: WSDM, 2020, pp. 735–743.
[40] J. Zhu, J. Liu, W. Li, J. Lai, X. He, L. Chen, Z. Zheng, Ensembled ctr prediction via knowledge distillation, in: CIKM, 2020, pp. 2941–2958.
[41] C. Xu, Q. Li, J. Ge, J. Gao, X. Yang, C. Pei, F. Sun, J. Wu, H. Sun, W. Ou, Privileged features distillation at taobao recommendations, in: SIGKDD, 2020, pp. 2590–2598.
[42] Y. Zhang, T. Xiang, T.M. Hospedales, H. Lu, Deep mutual learning, in: CVPR, 2018, pp. 4320–4328.
[43] Y. Wang, Y. Li, S. Li, W. Song, J. Fan, S. Gao, L. Ma, B. Cheng, X. Cai, S. Wang, et al., Deep graph mutual learning for cross-domain recommendation, in: DASFAA, Springer, 2022, pp. 298–305.
[44] X. Chen, Y. Zhang, H. Xu, Z. Qin, H. Zha, Adversarial distillation for efficient recommendation with external knowledge, TOIS 37 (1) (2018) 1–28.
[45] X. Liang, D. Li, A. Madden, Attributed network embedding based on mutual information estimation, in: CIKM, 2020, pp. 835–844.
[46] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, J. Tang, Session-based social recommendation via dynamic graph attention networks, in: WSDM, 2019, pp. 555–563.
[47] R. He, J.J. McAuley, Ups and downs: modeling the visual evolution of fashion trends with one-class collaborative filtering, in: WWW, 2016, pp. 507–517.
[48] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T.-S. Chua, Neural collaborative filtering, in: WWW, 2017, pp. 173–182.
[49] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, BPR: Bayesian personalized ranking from implicit feedback, in: UAI, 2009, pp. 452–461.