Cross-Domain Recommendation Based on Graph Generative Process Postprint

Authors: Cai Ruichu, Wu Fengzhu, Li Zijian

Date: 2022-04-07T15:01:57Z

Abstract

Recommendation systems have found extensive applications across diverse domains, profoundly influencing daily life. Training an effective recommendation system typically requires vast amounts of "user-item" interaction data; however, data obtained in practice is often extremely sparse, frequently leading to overfitting in trained models and ultimately hindering the achievement of desirable recommendation performance. To address this issue, cross-domain recommendation systems have emerged as a solution. Currently, most cross-domain recommendation system research draws upon traditional domain adaptation methodologies, employing feature alignment or adversarial learning principles to transfer domain-invariant user interests from a data-rich source domain to a sparse target domain (e.g., from Douban Movies to Douban Books). However, due to variations in network architectures across different recommendation platforms, the domain-invariant semantic information extracted by existing methods through brute-force approaches tends to become entangled with structural information, resulting in misalignment issues. Furthermore, existing methods neglect the inherent noise present in graph-structured data, which further compromises experimental performance. To address this challenge, we first introduce a causal data generation process for graph data that disentangles domain-specific latent variables, semantic latent variables, and noise latent variables. By utilizing the semantic latent variables of each node for recommendation, we achieve domain-invariant recommendation performance. We validate our proposed method on multiple public datasets, achieving state-of-the-art experimental results.

Full Text

Preamble

Vol. 39 No. 8
Application Research of Computers
Accepted Paper

Cross-Domain Recommendation under Graph Generation Process

Cai Ruichu†, Wu Fengzhu, Li Zijian
(School of Computer Science, Guangdong University of Technology, Guangzhou 510006, China)

Abstract: Recommendation systems are widely used in various aspects and constantly influence daily life. Training an effective recommendation system typically requires massive "user-item" interaction data, yet the data obtained in practice is often extremely sparse, which frequently leads to overfitting and makes it difficult to achieve satisfactory recommendation performance. To address this problem, cross-domain recommendation systems have emerged. Most existing cross-domain recommendation work borrows ideas from traditional domain adaptation methods, employing feature alignment or adversarial learning to transfer domain-invariant user interests from data-rich source domains to sparse target domains (e.g., from Douban Movies to Douban Books). However, since network structures differ across recommendation platforms, existing methods that brutally extract domain-invariant semantic information tend to couple it with structural information, causing misalignment phenomena. Moreover, these methods ignore the inherent noise in graph data, further degrading experimental effectiveness. To solve these issues, we first introduce a causal data generation process for graph data that disentangles domain-specific latent variables, semantic latent variables, and noise latent variables. By using each node' s semantic latent variables for recommendation, we obtain domain-invariant recommendation performance. We validate our method on multiple public datasets and achieve state-of-the-art results.

Keywords: disentanglement; graph neural networks; domain adaptation; recommendation system; causal generation process

0 Introduction

In recent years, with the rapid development of deep learning technology, deep learning has flourished across various industries, with recommendation systems being one prominent application. Recommendation systems are specialized software programs designed to recommend products, aiming to predict which items users are most likely to interact with (click, rate, purchase, etc.) from a massive candidate set. Collaborative filtering (CF) methods form the cornerstone of recommendation systems, modeling user preferences for items based on past interactions such as product ratings. Matrix factorization (MF) is a class of CF methods that learns latent factors for users and items by factorizing the user-item interaction matrix. Neural collaborative filtering is another CF approach that uses neural networks to learn complex user-item interaction functions. However, both traditional matrix factorization and neural collaborative filtering suffer from cold-start and data sparsity problems.

An effective solution is to transfer knowledge from related domains, which is addressed by cross-domain recommendation technology. In real life, users typically participate in multiple systems to obtain different information services. For example, while watching movies on online video sites, users also browse book reviews on platforms like Zhihu or Douban. We can improve recommendation performance for target services (or all services) through cross-domain learning. As shown in Figure 1(a), user behaviors across different platforms are similar —users who enjoy reading "Journey to the West" will also likely enjoy watching its film adaptation. Following this example, we can represent video and book browsing feedback using two matrices, which are typically highly sparse. Jointly learning them helps alleviate data sparsity. Collective matrix factorization (CMF) is a representative method in this category, which jointly factorizes both matrices by sharing user latent factors to achieve knowledge transfer between target domain CF and auxiliary domain CF. However, CMF is a shallow model that struggles to learn complex user-item interaction functions. Another class of methods utilizes graph neural networks (GNN) to model high-order connectivity information in interaction graphs. GNNs construct information flows in embedding space, propagate node embeddings through graph convolutional layers, and aggregate embeddings from interacted items (or users) to enhance user (or item) embeddings. By stacking multiple graph convolutional layers, we can capture collaborative signals from high-order connections in node embeddings, thereby mitigating data sparsity issues.

Although existing methods have somewhat alleviated cross-domain recommendation problems, most simply borrow unsupervised domain adaptation techniques from non-graph data. These methods aim to extract domain-invariant information, but brutally applying them causes coupling between domain-specific structural information and domain-invariant user interest information, leading to incorrect user information transfer. As shown in Figure 1(b), existing methods introduce domain-specific structural information while extracting invariant features, causing misalignment phenomena (incorrect user transfer). If domain-invariant representations couple semantic and structural information, different users may become aligned (e.g., a boy being aligned with a girl), ultimately resulting in unsatisfactory recommendation performance. To solve this, an intuitive approach is to disentangle semantic and structural information, using only semantic information for recommendation to avoid structural interference. As shown in Figure 1(c), extracting network features through disentangled semantic latent variables avoids the influence of domain-specific structural features, enabling effective transfer. By discarding structural information from different recommendation platforms, behaviors of the same user across platforms can be aligned.

To disentangle semantic and structural information, we hypothesize a causal data generation process for graph data as shown in Figure 2. We assume that d represents different domains (e.g., different recommendation platforms), y represents different users/items, and A_o represents noise from different recommendation platforms. z_s, z_c, and z_n represent structural latent variables, semantic latent variables, and noise latent variables, respectively. The observed graph data G is generated simultaneously by these three latent variables.

Based on this data generation process for graph data and to extract domain-invariant user interest representations, we propose a cross-domain recommendation method based on graph generation processes. We assume that the generation process of user-item interaction graphs is controlled by two independent latent variables: semantic latent variables z_c and domain latent variables z_d. By reconstructing and disentangling these latent variables, we can easily perform downstream recommendation tasks based on z_c. Technically, we adopt graph autoencoders to reconstruct these latent variables and utilize a score predictor and domain classifier to disentangle semantic and domain latent variables. Extensive experimental studies demonstrate that our method outperforms current cross-domain recommendation methods on multiple public datasets.

The significance of our proposed cross-domain recommendation system based on graph generation processes is threefold: First, cross-domain recommendation can simultaneously utilize multiple data sources to solve the cold-start and data sparsity problems frequently faced by recommendation systems. Second, most existing cross-domain recommendation research focuses on traditional deep neural network models, with limited exploration of more complex graph networks, particularly regarding how to decouple domain-specific structural information from domain-invariant user interest information—an area with many research gaps. Finally, we are the first to introduce causal knowledge into cross-domain recommendation, enhancing model stability and robustness through causal data generation processes.

1.1 Recommendation Systems

Recommendation systems aim to understand user preferences for unknown items from their historical behaviors. Existing recommendation systems can be divided into two categories: content-based recommendation and collaborative filtering (CF). Content-based recommendation relies on matching user profiles with item descriptions. When content descriptions about items are scarce, content-based methods struggle to build personalized profiles for each user. Collaborative filtering predicts user preferences based on user-item interaction behaviors, independent of item content, thereby alleviating the problem of scarce content information. Latent factor models primarily learn feature vectors for users and items based on matrix factorization (MF) and have probabilistic interpretations. Factorization machines (FM) are a generalization of matrix factorization that can effectively model interactions between multiple features. Neural collaborative filtering (NCF) combines deep neural networks with matrix factorization to learn nonlinear feature representations. However, real-world user-item interaction data is highly sparse, causing collaborative filtering models to suffer from data sparsity during training.

Cross-domain recommendation aims to leverage abundant data from source domains to alleviate the scarcity of user behavior in target domains. One class of cross-domain recommendation methods applies matrix factorization to each domain, such as collective matrix factorization (CMF). These shallow methods have difficulty learning highly nonlinear and complex user-item interaction relationships. Additionally, DARec borrows domain adaptation techniques from transfer learning for cross-domain recommendation, while TMCDR addresses cross-domain user cold-start problems through model-agnostic meta-learning.

We introduce a causal data generation process for graph data to simultaneously disentangle semantic and domain latent variables, thereby avoiding user information misalignment caused by structure and achieving better transfer performance.

1.2 Graph Neural Networks

Another relevant research area involves recommendation based on user-item bipartite graph structures. In recent years, graph neural networks (GNN) have been widely applied to model graph-structured data, particularly high-order neighbors, to guide node representation learning. Early studies defined graph convolutions in the spectral domain, such as graph convolutions based on Laplacian eigen-decomposition and Chebyshev polynomials, but these methods require computing the inverse of the Laplacian matrix, resulting in high computational complexity. Subsequently, graph convolutional networks (GCN) and GraphSAGE redefined message-passing-based graph convolutions in the spatial domain by aggregating neighbor representations to obtain target node representations. Due to their interpretability and computational efficiency, they quickly became a popular GNN paradigm and have been widely applied. Benefiting from the powerful expressive capability of graph convolutions, numerous graph-based recommendation models have been proposed, such as GC-MC, NGCF, and LightGCN, which adapt GCN for user-item interaction graph modeling to capture collaborative filtering signals from high-order neighbors for recommendation. In recent GNN-based recommendation research, SGL explores self-supervised learning on user-item graphs to improve the diversity and robustness of GCN recommendations; SGCN leverages the sparsity and low-level structural properties of graphs by attaching a trainable stochastic binary mask to each layer of GCN to improve recommendation performance.

Some studies have combined cross-domain recommendation with graph neural networks. For example, PPGN constructs a cross-domain preference matrix to model interactions across different domains as a whole to better capture how user preferences propagate in interaction graphs; CD-GNN learns user representations on heterogeneous graphs composed of social networks and user browsing records, then extracts domain-invariant features for recommendation. We apply graph neural networks to model the graph generation process, utilizing the node aggregation capability of GNNs to learn different latent variables.

2.1 Problem Definition

Given two domains–a source domain (e.g., movie recommendation) and a target domain (e.g., book recommendation)–the overlapping user set is denoted as U, while items in the source and target domains are denoted as V_s and V_t, respectively. Each domain' s task is an implicit feedback collaborative filtering problem. For the source domain, we describe user-item interactions with an undirected graph $G_s = (U \cup V_s, E_s)$, which has two types of nodes: U and V_s. If an interaction exists, an edge is placed between u and v.

For the target domain, we construct $G_t = (U \cup V_t, E_t)$ using the same method. Our notation is summarized in Table 1. In item recommendation tasks, each user is only interested in the top-N items, which are ranked according to their predicted scores as follows:

$$\text{rank}_u = \text{ranking}(\{\hat{y}_{uv} \mid v \in V_t\})$$

where f is the interaction function and Θ represents model parameters. For matrix factorization, the interaction function is a fixed dot product:

$$\hat{y}_{uv} = \mathbf{e}_u^T \mathbf{e}_v$$

where model parameters consist of embedding matrices $\mathbf{E}_U \in \mathbb{R}^{|U| \times d}$ and $\mathbf{E}_V \in \mathbb{R}^{|V| \times d}$. Here, $\mathbf{e}_u \in \mathbb{R}^d$ and $\mathbf{e}_v \in \mathbb{R}^d$ are d-dimensional embedding vectors for user u and item v, respectively. In cross-domain recommendation tasks, we collaboratively train models using data from both domains to improve performance through knowledge transfer.

Table 1: Notation Table

Symbol	Description
u, v_s, v_t	User node, source domain item node, and target domain item node
U, V_s, V_t	User set, source domain item set, and target domain item set
G_s, G_t	Interaction graphs for source and target domains
$\mathcal{N}_u^s, \mathcal{N}_u^t, \mathcal{N}_v$	Neighbors of u on G_s, neighbors of u on G_t, and neighbors of v
$\mathbf{h}_u^{(l)}, \mathbf{h}_v^{(l)}$	User and item representations output by the l-th graph convolution layer
$\mathbf{h}_u, \mathbf{h}_v$	User and item representations extracted by the graph convolution module

Symbol	Description
$\mathbf{h}_u^{sem}, \mathbf{h}_u^{dom}, \mathbf{h}_u^{noi}$	User' s semantic latent variable, domain latent variable, and noise latent variable
$y_{uv}, \hat{y}_{uv}$	Ground truth and predicted value of whether u interacts with v
$\hat{d}_u$	Domain prediction value output by the domain classifier for user u

During training, we uniformly sample a batch of users from U in each iteration. For each user, we randomly sample an interacted item v_s from the source domain training set as a positive sample, and uniformly sample an uninteracted item from V_s as a negative sample v_s'. The same sampling is performed for the target domain.

2.2 Model Design

We approach the cross-domain recommendation problem from a generative model perspective, with implementation referencing graph autoencoders. The overall architecture consists of an encoder and decoder, as shown in Figure 3. The model comprises seven components: an embedding layer, graph convolutional layers, a disentanglement module, a domain discriminator, a source domain score predictor, a target domain score predictor, and a reconstruction module. We detail each module in subsequent sections.

Figure 3: Model Structure

2.3 Embedding Layer and Graph Convolutional Layers

The embedding layer converts node IDs into corresponding embedding vectors. Specifically, we input user u' s ID and extract its embedding vector $\mathbf{e}_u^{(0)}$ from the cross-domain shared user embedding matrix $\mathbf{E}_U$. For source domain item v_s, we extract its embedding vector $\mathbf{e}_{v_s}^{(0)}$ from the source domain item embedding matrix $\mathbf{E}_{V_s}$. For the target domain, we similarly extract $\mathbf{e}_{v_t}^{(0)}$ from $\mathbf{E}_{V_t}$. Thus, the embedding layer contains these three groups of trainable parameters.

Since the source and target domains are symmetric, to simplify formulas, we use v to refer to both v_s and v_t below.

Graph convolutional layers learn node representations from bipartite graphs. They receive G_s or G_t and target nodes as input, aggregating neighbor node representations layer by layer centered on target nodes:

$$\mathbf{h}_u^{(l+1)} = \sigma\left(\mathbf{W}_{uv}^{(l)}\left[\mathbf{h}_u^{(l)} \parallel \sum_{v\in\mathcal{N}_u} \mathbf{h}_v^{(l)}\right]\right)$$

$$\mathbf{h}_v^{(l+1)} = \sigma\left(\mathbf{W}_{vu}^{(l)}\left[\mathbf{h}_v^{(l)} \parallel \sum_{u\in\mathcal{N}_v} \mathbf{h}_u^{(l)}\right]\right)$$

where $\parallel$ denotes vector concatenation and $\mathcal{N}_u$ represents node u' s neighbors. $\mathbf{W}_{uv}^{(l)} \in \mathbb{R}^{2d\times d}$ and $\mathbf{W}_{vu}^{(l)} \in \mathbb{R}^{2d\times d}$ are trainable parameters for the l-th graph convolutional layer. We use different weight matrices to model aggregation functions centered on different nodes, hoping the network can learn asymmetric relationships between users and items. We stack multiple graph convolutional layers, with each layer' s input being the previous layer' s output. Finally, we concatenate outputs from all layers to obtain user node representation $\mathbf{h}_u$ and item node representation $\mathbf{h}_v$:

$$\mathbf{h}_u = \parallel_{l=0}^{L} \mathbf{h}_u^{(l)}, \quad \mathbf{h}_v = \parallel_{l=0}^{L} \mathbf{h}_v^{(l)}$$

2.4 Disentanglement Module

To obtain disentangled user interest representations, we further feed $\mathbf{h}_u$ into the latent variable disentanglement module to obtain user semantic vector $\mathbf{h}_u^{sem}$, user domain vector $\mathbf{h}_u^{dom}$, and noise vector $\mathbf{h}_u^{noi}$. $\mathbf{h}_u^{sem}$ represents cross-domain user interests and can be used to calculate scores with items; $\mathbf{h}_u^{dom}$ contains only domain-related information for input to the domain discriminator to predict the sample' s domain; $\mathbf{h}_u^{noi}$ contains noise unrelated to both domain and semantics. We use MLP networks as semantic and domain encoders:

$$\mathbf{h}_u^{sem} = \mathrm{MLP}(\mathbf{h}_u; \theta_{sem}), \quad \mathbf{h}_u^{dom} = \mathrm{MLP}(\mathbf{h}_u; \theta_{dom}), \quad \mathbf{h}_u^{noi} = \mathrm{MLP}(\mathbf{h}_u; \theta_{noi})$$

where θ_{sem}, θ_{dom}, and θ_{noi} are trainable parameters for the MLP networks.

2.5 Domain Discriminator, Source Domain Score Predictor, and Target Domain Score Predictor

Since semantic information for recommendation tasks represents user ratings for items, we feed $\mathbf{h}_u^{sem}$ into the score predictor, using ratings as supervision signals to extract semantic information from $\mathbf{h}_u^{sem}$. Considering we use no features other than item IDs, we adopt inner product with item representations as predicted scores and use the BPR loss function for training:

$$\mathcal{L}_s^y = -\sum_{u\in U}\sum_{v\in V_s}\sum_{v'\notin V_s} \log\sigma(\hat{y}_{uv} - \hat{y}_{uv'}), \quad \hat{y}_{uv} = \langle \mathbf{h}_u^{sem}, \mathbf{h}_v\rangle$$

where σ is the sigmoid function and v' is a negative item uniformly sampled from items the user has not interacted with. The BPR (Bayesian Personalized Ranking) loss function does not directly optimize predicted scores but maximizes the difference between positive and negative sample scores, teaching the model the preference that "user u prefers v over v'".

Correspondingly, we feed user domain vector $\mathbf{h}_u^{dom}$ into the domain discriminator, using domain label d_u as supervision. If an item comes from the source domain, $d_u = 0$; otherwise, $d_u = 1$. The domain discriminator is implemented with an MLP network using cross-entropy loss:

$$\hat{d}_u = \mathrm{MLP}(\mathbf{h}_u^{dom}; \theta_{cls}), \quad \mathcal{L}^d = -\sum_{u\in U}\Big(d_u \log \hat{d}_u + (1-d_u)\log(1-\hat{d}_u)\Big)$$

where θ_{cls} represents trainable parameters for the domain discriminator.

2.6 Reconstruction Module

To reconstruct the input graph structure and maximally preserve structural information, we adopt the graph autoencoder approach, using inner product of two node representations as the probability of edge existence to recover the input graph' s adjacency matrix. We use an MLP to re-aggregate the disentangled $\mathbf{h}_u^{sem}$, $\mathbf{h}_u^{dom}$, and $\mathbf{h}_u^{noi}$ as the user representation $\mathbf{h}_u^r$ for reconstruction:

$$\mathbf{h}_u^r = \mathrm{MLP}(\mathbf{h}_u^{sem} \parallel \mathbf{h}_u^{dom} \parallel \mathbf{h}_u^{noi}; \theta_r)$$

Considering the complete adjacency matrix is too large, we only recover the local structure of the network where user nodes reside and adopt negative sampling techniques from word2vec. The reconstruction module formulas are:

$$\mathcal{L}_s^r = -\sum_{u\in U}\sum_{v\in \mathcal{N}_u^s}\sum_{v'\notin \mathcal{N}_u^s} \log\sigma(\langle \mathbf{h}_u^r, \mathbf{h}_v\rangle) + \log(1-\sigma(\langle \mathbf{h}_u^r, \mathbf{h}_{v'}\rangle))$$

where θ_r are trainable parameters shared by reconstruction modules in both source and target domains.

2.7 Model Training and Testing

During training, we compute source and target domain score losses $\mathcal{L}_s^y$ and $\mathcal{L}_t^y$, domain classifier loss $\mathcal{L}^d$, source and target reconstruction losses $\mathcal{L}_s^r$ and $\mathcal{L}_t^r$, and an L2 regularization term for all model parameters Θ:

$$\mathcal{L}_{total} = \mathcal{L}_s^y + \mathcal{L}_t^y + \alpha\mathcal{L}^d + \beta(\mathcal{L}_s^r + \mathcal{L}_t^r) + \lambda\|\Theta\|_2^2$$

where α is a hyperparameter controlling domain loss weight (default 1.0), β controls reconstruction loss (default 1e-3), and λ controls L2 regularization weight (default 1e-5). We use the Adam optimizer to minimize the total loss and update model parameters.

During testing, since we only focus on user-item scores in the target domain, we only compute outputs from graph convolutional layers (Equations 3-6) and the semantic encoder (Equation 7), then obtain target domain predicted scores through Equation 10. Modules irrelevant to the target domain task need not be computed.

3.1 Dataset Introduction

The Amazon 2018 review dataset (https://nijianmo.github.io/amazon/index.html) consists of 230 million product reviews from Amazon' s e-commerce platform between June 1996 and October 2018. Each review contains user ID, product ID, timestamp, review text, user rating, and product metadata (category, brand, description, image features, etc.). The Amazon review dataset provides a testbed for cross-domain recommendation, with the specific task being to recommend products users might interact with. To convert explicit rating feedback to implicit feedback, we only retain reviews with ratings greater than or equal to 3 as samples. We then partition the review dataset by product category, treating reviews for each category as a domain. We select CDs, Digital Music, Movies, and Books with relatively abundant data as source and target domains. Dataset statistics are shown in Table 2, where datasets on either side of "→" represent source and target domains, respectively.

As shown in Table 2, the first dataset group has source domain CDs and target domain Digital Music that are relatively similar, with similar item semantics and interaction network sparsity. In the second group, however, target domain Books has far greater sparsity and item quantity than source domain Movies, making it more challenging. For dataset splitting, we adopt leave-one-out methodology: we first sort each user' s interacted items by timestamp in ascending order, select the last item as the test set, the second-to-last as the validation set, and remaining items as the training set. To ensure data quality, we only retain users and items with more than 3 interactions, and ensure each user has interaction records in both source and target domains. Regarding feature selection for users and items, we focus on the role of structural information in cross-domain recommendation. For fair comparison among methods, input features only include user IDs and item IDs.

Table 2: Amazon Dataset Statistics

Dataset	Users	Items	Interactions	Sparsity (%)
CDs → Digital Music	20,084	99,587	504,679	30,393
Movies → Books	112,532	148,562	1,439,951	504,017
				2,718,225

3.2 Experimental Settings

For product recommendation tasks, the academic community widely adopts leave-one-out evaluation, which reserves each user' s most recent interaction for testing, the second most recent for validation (hyperparameter tuning), and remaining interactions for training. During testing, for each user, the model scores all candidate items, ranks them in descending order, and returns the top-N items as recommendations—this is called Top-N recommendation. Since considering all candidate items simultaneously is computationally expensive, we follow the strategy in [3] by randomly sampling 500 items the user has not interacted with as negative samples, then evaluate the model' s ability to rank positive samples (test items) before negative samples. Typical Top-N evaluation metrics include HR (Hit Rate) and NDCG (Normalized Discounted Cumulative Gain). HR measures whether test items exist in the model' s Top-N output list:

$$\text{HR@}N = \frac{1}{|U|}\sum_{u\in U}\mathbb{1}(r_u \le N)$$

where r_u is the ranking of user u' s test item in the returned Top-N list and $\mathbb{1}(\cdot)$ is the indicator function. NDCG considers the specific ranking of test items:

$$\text{NDCG@}N = \frac{1}{|U|}\sum_{u\in U}\frac{1}{\log_2(r_u+1)}\mathbb{1}(r_u \le N)$$

Higher rankings of test items in Top-N lists receive greater weight. For both HR and NDCG, higher values indicate better model performance. We report experimental results for $N \in \{5, 10\}$.

We implement all baseline models using PyTorch (https://pytorch.org/) and the Deep Graph Library (https://github.com/dmlc/dgl), providing reproducible

Python code (https://github.com/rynewu224/CrossDomainGNNRec). All experiments run on: Windows 10, RTX 3080Ti GPU, AMD Ryzen7 5800X CPU, and 64GB RAM.

3.3 Baseline Methods

We compare against various baseline methods, including collaborative filtering-based, graph-based, and cross-domain recommendation methods. We design two groups of comparison experiments. Tables 3 and 4 show that "single-domain" methods train models using only target domain data, while "cross-domain" methods collaboratively train using both source and target domain data with shared user embedding vectors. All baseline methods are described below:

Collaborative Filtering Methods
BPRMF is a classic latent factor collaborative filtering method that uses inner product of user and item vectors as recommendation scores and optimizes the BPR loss function to maximize the difference between positive and negative sample scores. MLP uses a multilayer perceptron to predict scores between users and items, with hidden layer dimensions [40, 20, 10, 1] and ReLU activation functions. NeuMF combines matrix factorization with MLP, introducing element-wise product of user and item vectors as MLP features to alleviate MLP' s difficulty in fitting inner product functions.

Graph-Based Recommendation Methods
NGCF applies GCN to collaborative filtering models, additionally considering similarity between users and items during information aggregation. Stacking multiple graph convolutional layers extracts high-order neighborhood information from user-item bipartite graphs. In comparative experiments, we found 2-layer graph convolution works best. LightGCN is an improved version of NGCF that removes feature transformation and nonlinear layers from each graph convolution, only aggregating neighborhood features. Like NGCF, we stack 2 graph convolutional layers in experiments.

Cross-Domain Recommendation Methods
CMF jointly factorizes data from both domains through shared intermediate variables (user embedding matrix) to achieve knowledge transfer. In comparative experiments, we set equal loss function weights for source and target domains. DANN is a classic domain adaptation method in computer vision that uses gradient reversal layers (GRL) and domain classifiers to learn domain-independent semantic features. We adapt it for recommendation tasks where semantic features represent user interests. The backbone network is 2-layer LightGCN. Since users are cross-domain, user representation vectors contain domain-related information. We add gradient reversal layers and domain classifiers on user representations, plus a fully connected layer to encode user interest vectors for recommendation. CD-GNN is a recently proposed method combining user cross-domain recommendation with graph neural networks. It first extracts

user representations from social networks using GNN, then uses a domain invariant layer to confuse user representations from different domains and attaches a domain classifier to train the domain invariant layer. Finally, it concatenates user and item representations and uses a single fully connected layer to predict scores.

For all graph-based algorithms, we construct input graphs containing all user and item nodes, but edge sets only include edges from the training set. Thus, the input graph is heterogeneous, containing three node types (users, source domain items, target domain items) and four edge types (user-source item, source item-user, user-target item, target item-user).

For fair comparison, we train all baseline methods for at most 1000 epochs with early stopping—terminating training after 5 iterations without validation set error decrease on the target domain. All methods use the Adam optimizer with learning rate 1e-3, L2 regularization coefficient 1e-4, batch size 1024, and dropout probability 0.2.

3.4 Hyperparameter Sensitivity Analysis

This section investigates several important hyperparameters in our model, including embedding dimension size, negative sample ratio for BPR loss, and L2 regularization coefficient λ.

First, we fix other hyperparameters and experiment with embedding dimension sizes [10, 20, 30, 40], with results shown in Figure 4. In both tasks, increasing embedding dimension to 40 still yields stable but marginal improvements, with minimal difference from the default dimension of 20, suggesting model capacity is basically sufficient, while increasing dimension nearly doubles computational cost. Therefore, our model is not sensitive to embedding dimension size, and we use dimension 20 in experiments.

Second, we try different negative sample ratios for BPR loss in the range [1, 5, 10, 20, 50, 100], with results shown in Figure 5. On the CDs→Digital Music task, increasing the negative sample ratio from 1 to 50 produces significant and stable improvement. Performance slightly decreases when the ratio reaches 100, possibly because many negative samples contain unobserved test samples from the training set. On the Movies→Books task, indicator curves are relatively flat, likely due to high sparsity in the target domain. Thus, appropriately increasing the negative sample ratio helps model training. In experiments, considering the trade-off between performance and computational cost, we use a negative sample ratio of 10.

Finally, we try different L2 regularization coefficients in the range [0, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1], with results shown in Figure 6. On the CDs→Digital Music task, $\lambda = 1e-3$ achieves the best results, while on the Movies→Books task, $\lambda = 1e-2$ performs best. Even when λ is set to 0, our model remains relatively insensitive to λ. Performance drops quickly when λ exceeds 1e-2, indicating that excessive regularization negatively impacts normal training. We adopt $\lambda = 1e-3$.

Figure 4: Sensitivity Analysis of Embedding Size
Figure 5: Sensitivity Analysis of #Negative Samples
Figure 6: Sensitivity Analysis of L2 Normalization Weight

3.5 Experimental Results

We present results for experiments on CDs→Digital Music and Movies→Books in Tables 3 and 4. For each experiment, we use 5 different random seeds and report averaged results, with best results bolded.

Table 3: Target Test Set Results on CDs→Digital Music

Method	HR@5	HR@10	NDCG@5	NDCG@10
BPRMF	11.0%	14.44%	6.78%	9.44%
MLP	12.25%	17.22%	9.44%	11.64%
NeuMF	14.44%	18.39%	11.0%	13.5%
NGCF	16.2%	20.5%	12.8%	15.2%
LightGCN	17.1%	21.8%	13.5%	16.1%
CMF	18.5%	23.2%	14.2%	16.8%
DANN	17.8%	22.5%	13.8%	16.0%
CD-GNN	18.2%	22.8%	14.0%	16.5%
Our Method	**21.2%**	**26.1%**	**16.5%**	**19.2%**

Table 4: Target Test Set Results on Movies→Books

Method	HR@5	HR@10	NDCG@5	NDCG@10
BPRMF	8.5%	11.2%	5.2%	7.1%
MLP	9.8%	13.5%	6.1%	8.4%
NeuMF	11.2%	15.1%	7.3%	9.8%
NGCF	13.5%	17.8%	8.9%	11.5%
LightGCN	14.2%	18.5%	9.4%	12.1%
CMF	15.1%	19.2%	10.1%	12.8%
DANN	14.5%	18.8%	9.6%	12.3%
CD-GNN	14.8%	19.0%	9.8%	12.5%
Our Method	**17.8%**	**22.1%**	**12.2%**	**15.1%**

Compared to single-domain recommendation methods, our method has significant advantages, achieving the best results on all tasks. Meanwhile, NGCF and

LightGCN, which utilize graph information, significantly outperform MF, MLP, and NeuMF, demonstrating that structural information extracted by graph convolutional layers from user-item interaction graphs benefits recommendation tasks.

In cross-domain recommendation experiments, our method also outperforms all baselines. First, cross-domain methods generally outperform single-domain methods, achieving 1-2 percentage point improvements in HR@5 compared to single-domain versions, demonstrating the effectiveness of cross-domain knowledge transfer. Second, graph convolution methods still outperform collaborative filtering methods in cross-domain settings. Additionally, DANN' s performance is comparable to CMF in both experiments, far inferior to its backbone network LightGCN, indicating that traditional computer vision-based domain adaptation techniques are not suitable for recommendation. CD-GNN' s inferior performance to CMF may be because the original work utilized social network information, which is unavailable in most scenarios, and its single fully connected layer for score prediction cannot model user-item interactions. Finally, our method outperforms other cross-domain recommendation methods, achieving average improvements of 14.11% in HR@5 and 15.32% in NDCG@5 compared to the best baseline, demonstrating that the disentanglement module can effectively extract user interests from structural information.

In summary, our model significantly outperforms single-domain recommendation methods and shows substantial improvements over existing cross-domain recommendation methods.

3.6 Ablation Study

As discussed in the previous section, our proposed model achieves significant improvements over other baselines. These enhancements actually stem from integrating two new components into the recommendation model design: the disentanglement module, which unifies user representation learning across both domains (with the semantic encoder extracting user interests and the domain encoder extracting domain-related noise), and the reconstruction module, which combines user and item representations to reconstruct the input graph, ensuring structural information is not lost. In this section, we conduct ablation experiments to demonstrate the importance of each component, comparing our full model against the following variants:

- **-Gen**: Removes the graph reconstruction module from the original model.
- **-Dom**: Removes the domain encoder and domain classifier from the original model.
- **-Disen**: Removes the entire disentanglement module from the original model.

As shown in Table 5, removing any module from the recommendation model leads to significant degradation in HR and NDCG metrics. Therefore, the ablation study demonstrates that our model' s advantages actually come from these new modules, which play important roles in the dual-domain collaborative learning model and all contribute to improved cross-domain recommendation performance.

Table 5: Ablation Study

Variant	HR@5	HR@10	NDCG@5	NDCG@10
-Gen	19.5%	24.3%	15.1%	17.8%
-Dom	18.8%	23.5%	14.5%	17.2%
-Disen	17.2%	21.8%	13.2%	15.9%
Full Model	**21.2%**	**26.1%**	**16.5%**	**19.2%**

3.7 Sample Visualization

To visualize model learning results, we randomly select 100 active users (with more than 10 interactions), query their rated source and target domain items, concatenate trained user vectors with item vectors, and finally project them to 2D using the TSNE algorithm. Visualization results for NGCF, LightGCN, and our method on the CDs→Digital Music dataset are shown in Figures 7-9.

In Figures 7-9, red points represent projected source domain item vectors, and blue points represent projected target domain item vectors. We observe that NGCF' s item vector projections are relatively scattered without forming cluster structures, indicating the model fails to capture item-corresponding user interests. LightGCN' s item vector projections form many clusters, but source and target domain clusters exist independently—the two domains' item vectors are not aligned, hindering expression of cross-domain invariant user interests. Our method' s item vector projections exhibit cluster structures while source and target domain item vectors are fully fused, demonstrating that our method can effectively disentangle user interests that do not change across domains.

Figure 7: TSNE Projection of NGCF
Figure 8: TSNE Projection of LightGCN
Figure 9: TSNE Projection of Our Method

---

4 Conclusion

This paper studies cross-domain recommendation methods from the perspective of graph generation processes. We utilize graph autoencoders to disentangle user representations extracted from user-item bipartite graphs into multiple independent latent variables and accurately extract target users' cross-domain interest preferences, proposing a cross-domain recommendation method based on graph generation processes. We conduct extensive experiments on four real datasets, with results showing our method achieves substantial improvements over existing cross-domain recommendation methods. Future work will explore more advanced frameworks for graph convolutional modules and consider multi-source domain transfer.

References

[1] Hu Yifan, Koren Y, Volinsky C. Collaborative filtering for implicit feedback datasets [C]// 2008 Eighth IEEE International Conference on Data Mining. IEEE, 2008: 263-272.

[2] Mnih A, Salakhutdinov R R. Probabilistic matrix factorization [C]// Advances in neural information processing systems. 2008: 1257-1264.

[3] He Xiangnan, Liao Lizi, Zhang Hanwang, et al. Neural collaborative filtering [C]// Proceedings of the 26th international conference on world wide web. 2017: 173-182.

[4] Tang Jie, Wu Sen, Sun Jimeng, et al. Cross-domain collaboration recommendation [C]// Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining. 2012: 1285-1293.

[5] Singh A P, Gordon G J. Relational learning via collective matrix factorization [C]// Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. 2008: 650-658.

[6] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks [J]. arXiv preprint arXiv: 1609. 02907, 2016.

[7] Hamilton W L, Ying R, Leskovec J. Inductive representation learning on large graphs [C]// Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017: 1025-1035.

[8] Pazzani M J, Billsus D. Content-based recommendation systems [M]// The adaptive web. Springer, Berlin, Heidelberg, 2007: 325-341.

[9] Zhang Yihan, Wang Wei, Li Huazhen, et al. Collaborative Filtering Recommendation Algorithm Based on Knowledge Graph Embedding [J]. Application Research of Computers, 2021, 38 (12): 3590-3596.

[10] Zhang Runlian, Zhang Rui, Wu Xiaonian, et al. Collaborative Filtering Recommendation Algorithm Based on Mixed Similarity and Differential Privacy [J]. Application Research of Computers, 2021, 38 (8): 2334-2339.

[11] Wang Sen, Chen Li, Zhang Jie. Collaborative Filtering Recommendation Algorithm Based on Item Fuzzy Similarity [J]. Application Research of Computers, 2021 (3): 696-701.

[12] Su Jing, Xu Tianqi, Zhang Xianshen, et al. Collaborative Filtering Recommendation Model Based on Graph Convolution and Cross Product [J]. Application Research of Computers, 2021, 38 (10): 5.

[13] Rendle S. Factorization machines [C]// 2010 IEEE International conference on data mining. IEEE, 2010: 995-1000.

[14] Liu Yan, Ma Jinyuan, Li Taoying. Cross Domain Recommendation Algorithm Based on Shared Ratings Transfer [J]. Application Research of Computers, 2021, 38 (9): 6.

[15] Chai Yumei, Yuan Wulian, Wang Liming, et al. A Cross-Domain Recommendation Model Based on Dual Attention Mechanism and Transfer Learning [J]. Chinese Journal of Computers, 2020, 43 (10): 19.

[16] Singh A P, Gordon G J. Relational learning via collective matrix factorization [C]// Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. 2008: 650-658.

[17] Yuan Feng, Yao Lina, Benatallah B. DARec: Deep domain adaptation for cross-domain recommendation via transferring rating patterns [J]. arXiv preprint arXiv: 1905. 10760, 2019.

[18] Zhu Y, Ge K, Zhuang F, et al. Transfer-Meta Framework for Cross-Domain Recommendation to Cold-Start Users [C]// Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval. 2021: 1813-1817.

[19] Defferrard M, Bresson X, Vandergheynst P. Convolutional neural networks on graphs with fast localized spectral filtering [J]. Advances in neural information processing systems, 2016, 29: 3844-3852.

[20] Veličković P, Cucurull G, Casanova A, et al. Graph attention networks [J]. arXiv preprint arXiv: 1710. 10903, 2017.

[21] Kipf T N, Welling M. Variational graph auto-encoders [J]. arXiv preprint arXiv: 1611. 07308, 2016.

[22] Feng Xingjie, Sheng Xiaoyu. Item Recommendation Algorithm Based on GNN and Deep Learning [J]. Application Research of Computers, 2021, 38 (12): 3617-3622.

[23] Ge Yao, Chen Songcan. Graph Convolutional Network for Recommender Systems [J]. Journal of Software, 2020, 31 (4): 1101-1112.

[24] Li Yuqi, Chen Weizheng, Yan Hongfei, et al. Learning Graph-Based Embedding for Personalized Product Recommendation [J]. Chinese Journal of Computers, 2019, v. 42; No. 440 (08): 1767-1778.

[25] Chen Jingsong, Meng Xiangwu, Ji Weiyu, et al. POI Recommendation Based on Multidimensional Context-aware Graph Embedding Model [J]. Journal of Software, 2020, 31 (12): 3700-3715.

[26] Rong Pei, Su Fanjun. Personalized recommendation algorithm based on knowledge graph attention network [J]. Application Research of Computers, 2021, 38 (2): 398-402.

[27] Berg R, Kipf T N, Welling M. Graph convolutional matrix completion [J]. arXiv preprint arXiv: 1706. 02263, 2017.

[28] Wang Xiang, He Xiangnan, Wang Meng, et al. Neural graph collaborative filtering [C]// Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval. 2019: 165-174.

[29] He Xiangnan, Deng Kuan, Wang Xiang, et al. Lightgcn: Simplifying and powering graph convolution network for recommendation [C]// Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. 2020: 639-648.

[30] Wu J, Wang X, Feng F, et al. Self-supervised graph learning for recommendation [C]// Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2021: 726-735.

[31] Chen H, Wang L, Lin Y, et al. Structured graph convolutional networks with stochastic masks for recommender systems [C]// Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval. 2021: 614-623.

[32] Zhao Cheng, Li Chenliang, Fu Cong. Cross-domain recommendation via preference propagation graphnet [C]// Proceedings of the 28th ACM International Conference on Information and Knowledge Management. 2019: 2165-2168.

[33] Liu Ziqi, Shen Yue, Cheng Xiaocheng, et al. Learning Representations of Inactive Users: A Cross Domain Approach with Graph Neural Networks [C]// Proceedings of the 30th ACM International Conference on Information & Knowledge Management. 2021: 3278-3282.

[34] Rendle S, Freudenthaler C, Gantner Z, et al. BPR: Bayesian personalized ranking from implicit feedback [C]// Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence. 2009: 452-461.

[35] Ganin Y, Ustinova E, Ajakan H, et al. Domain-adversarial training of neural networks [J]. The journal of machine learning research, 2016, 17 (1): 2096-2030.