Feature-Structure Adaptive Completion Graph Neural Network for Cold-start Recommendation

Songyuan Lei[1†], Xinglong Chang[1,2†], Zhizhi Yu[1*], Dongxiao He[1], Cuiying Huo[1], Jianrong Wang[1], Di Jin[1*]
[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]Qijia Youdao Network Technology (Beijing) Co., Ltd., Beijing, China
{leisongyuan, changxinglong, yuzhizhi, hedongxiao, huocuiying, wjr, jindi}@tju.edu.cn

Abstract

The cold-start recommendation has been challenging due to the limited historical interactions for new users and new items. Recently, methods based on meta-learning and graph neural networks have been effective in this problem. However, these methods mainly focus on the missing user-item interactions in cold-start scenarios, overlooking the missing of user/item feature information, which significantly limits the quality and effectiveness of node embeddings. To address this issue, we propose a new method called Feature-Structure Adaptive Completion Graph Neural Network (FS-GNN), which is designed to tackle the cold-start problem by simultaneously addressing the missing feature and structure information in a bipartite graph composed of users and items. Specifically, we first design a trainable feature completion module that leverages the knowledge emergence abilities of large language models to enhance node embedding and mitigate the impact of missing features. Then, we incorporate a three-channel structure completion module to simultaneously complete the structures among users-users, items-items, as well as users-items. Finally, we adaptively integrate the feature and structure completion modules in an end-to-end fashion, so as to minimize cross-module interference when completing features and structures simultaneously. This generates more comprehensive and robust embeddings for users and items in recommendation tasks. Experimental results on multiple public datasets demonstrate significant improvements in our proposed FS-GNN in cold-start scenarios, outperforming or being competitive with state-of-the-art methods.

Introduction

With the rapid development of online information, recommender system has become a key tool for alleviating users' information overload, especially on e-commerce and social media platforms. It mainly uses a personalized filtering mechanism (Covington, Adams, and Sargin 2016; Ying et al. 2018) to filter out the content that users are interested in from a massive amount of multimedia information including images, videos and audios, effectively improving the usability of information and user experience.

[†]These authors contributed equally.
[*]Corresponding authors.

Collaborative Filtering (CF) (Goldberg et al. 1992) techniques, especially the Matrix Factorization (MF) methods (Koren, Bell, and Volinsky 2009), have been widely applied in recommender systems across various domains due to their efficient predictive performance and scalability. However, these methods face challenges in handling sparse user-item interactions, especially when new users or items are involved. That is, the dreaded ”cold-start” problem, which leads to the degradation of recommendation performance. Specifically, on the one hand, from the perspective of structure, newly registered users and newly launched products typically lack sufficient historical interactions. Meanwhile, long-tail items that are seldom viewed or purchased also face a lack of interactions. These make it difficult for recommender systems to predict user preferences. On the other hand, from the perspective of feature, the issue of feature missing is widespread in recommender systems. For example, user information (e.g., nationality, age, and geographical location) may not be obtained due to privacy restrictions. Meanwhile, item information might also be incomplete, lacking detailed descriptions or classification tags. To this end, we categorize the feature missing problem into two types: user feature missing and item feature missing. The feature missing problem makes it difficult for recommender systems to fully utilize user/item feature information to enrich the representation of user/item nodes.

Several algorithms and models have been designed to handle the cold-start problem, including active learning (Zhu et al. 2020) or meta-learning (Dong et al. 2020; Lu, Fang, and Shi 2020). In particular, Graph Neural Networks (GNNs) (Kipf and Welling 2017) have been adopted to enrich user-item interactions due to their powerful ability to capture complex relational patterns and leverage the graph structure (He et al. 2020). For example, PGD (Wang et al. 2021) and MvDGAE (Zheng et al. 2021) introduce user/item features as nodes and form a heterogeneous graph in conjunction with the original user-item graph to enrich the final embeddings. MeGNN (Liu et al. 2023) explores enhancing meta-learning with GNN, which employs global neighborhood translation learning to achieve consistent interactions for new user and item nodes. However, these methods struggle to preserve feature similarity between user and item nodes and their feature nodes during propagation, exacerbating the over-smoothing phenomenon(Wang et al. 2023;

Dong et al. 2023). To sum up, the above methods only consider the missing of user-item interactions in cold-start recommendation and fail to consider the missing of user/item features. Considering this, AGNN (Qian et al. 2022) introduces a solution for the strict cold-start problem by leveraging feature information. While it points out the issue of missing features, it completes the missing features by manual crawling, which consumes a huge amount of cost.

In this paper, we consider the cold-start recommendation as a scenario where both the structure and feature are simultaneously missing in a graph. To this end, we employ graph completion methods to simultaneously complete both structure and feature information. However, there are two key challenges in using graph completion: (1) How to effectively leverage feature information for cold-start recommendation? Based on the assumption of homophily (McPherson, Smith-Lovin, and Cook 2001; Wang et al. 2022), user nodes with similar features often exhibit similar preferences, which might provide a basis for identifying the potential interests of new users (Covington, Adams, and Sargin 2016). (2) How to effectively alleviate the mutual influence between feature completion and structure completion? Existing graph completion methods tend to focus on either feature completion or structure completion (Jin et al. 2021; Chen et al. 2022). However, the presence of both incomplete features and structures may lead to unpredictable mismatches due to the randomness of missing data (Yang et al. 2022), which further exacerbates the interference between feature and structure completion, ultimately resulting in an incorrect completion.

To address the aforementioned challenges, we propose a novel Feature-Structure Adaptive Completion Graph Neural Network, namely FS-GNN, for cold-start recommendation. Specifically, inspired by large language models (LLMs) (Wei et al. 2023; Huang et al. 2024) and Graph Auto-Encoders (GAE) (Salehi and Davulcu 2020; Salha-Galvan et al. 2021), we first utilize the extensive knowledge of LLMs to complete features, and design a learnable feature completion module guided by the original graph to denoise and enhance the features. We then develop a three-channel structure completion module. That is, the first channel employs the Personalized PageRank (PPR) (Klicpera, Bojchevski, and Günnemann 2019) to calculate the associations between user and item nodes, completing the structure between highly associated node pairs to mitigate the missing interaction issue. The next two channels simultaneously utilize feature similarity to complete the structure between nodes of the same type, i.e., user-user and item-item relationships. Here we assume that users with similar features have similar preferences, and items with similar features attract similar user groups. Finally, we sequentially combine the feature and structure completion modules through an adaptive mechanism, avoiding mutual interference between the two. This effectively provides more comprehensive and robust node embeddings for the final rating prediction. Extensive experiments on three real-world datasets show that FS-GNN significantly outperforms state-of-the-art recommendation models in various cold-start scenarios, including meta-learning-based models (with a 12.44% improvement) and GNN-based models (with a 14.35% improvement).

Preliminaries

In this section, we first give the notations and then introduce the problem definition.

Notations. In collaborative filtering-based recommender systems, there are usually two types of entities: the set of users $U = \{u_1, u_2, \ldots, u_M\}$ and the set of items $I = \{i_1, i_2, \ldots, i_N\}$, where M and N represent the number of users and items, respectively. We treat the ratings provided by users to items as the interaction between them, and represent this with a rating matrix $\mathbf{R} \in \mathbb{R}^{M\times N}$, where $r_{ui} \in \mathbb{R}$ means that user u has a rating for item i, otherwise 0 indicates that the user has not yet interacted with this item. The user-item interactions can then be naturally formulated as a user-item bipartite graph: $G =< U \cup V, \mathbf{A} >$, where adjacency matrix $\mathbf{A}$ is constructed by the interaction matrix $\mathbf{R}$:

$$\mathbf{A} = \begin{bmatrix} \mathbf{0}^{M\times M} & \mathbf{R} \\ \mathbf{R}^T & \mathbf{0}^{N\times N} \end{bmatrix}, \tag{1}$$

In addition, each user and item has features from different fields, where each feature value is initialized by a universal discretization method (Qian et al. 2022). Then, all the features are concatenated into a multi-hot feature code. Here, we use $\mathbf{E}_u \in \mathbb{R}^{M\times D}$ and $\mathbf{E}_i \in \mathbb{R}^{N\times D}$ to denote the feature matrix of user nodes and item nodes, respectively.

Problem Definition. The goal of rating prediction is to predict a users' rating for an item. In the traditional warm-start scenario, we predict the rating of an item by a user with interaction history. While in cold-start scenarios, where there is a severe lack of interactions or even none for new users/items, our goal is to predict the ratings of new users/items that are not seen during training and have interactions only during the validation and testing phases. To evaluate the performance of the model, we split the recommendation task into three specific types to analyze the real-world scenarios in detail: (1) existing items for new users (User cold-start), (2) new items for existing users (Item cold-start), and (3) existing items for existing users (Warm-start).

In addition, the problem of missing features exists in many recommendation scenarios. This usually occurs when a new user or item provides only limited necessary information, or when the recommender systems fail to access private information. In this paper, we define it as two scenarios: (1) user features are missing and item features are complete (i.e., $u \in U^-, i \in I^+$), and (2) user features are complete and item features are missing (i.e., $u \in U^+, i \in I^-$).

Methodology

In this section, we first introduce the overall framework of the proposed FS-GNN. Then, we specify the modeling details of the feature completion and the structure completion. Finally, we introduce the objective function to be optimized for the entire model.

Overview

The core idea of our proposed approach is to view the cold-start problem as a problem where both features and structures are missing from a graph, and use graph completion

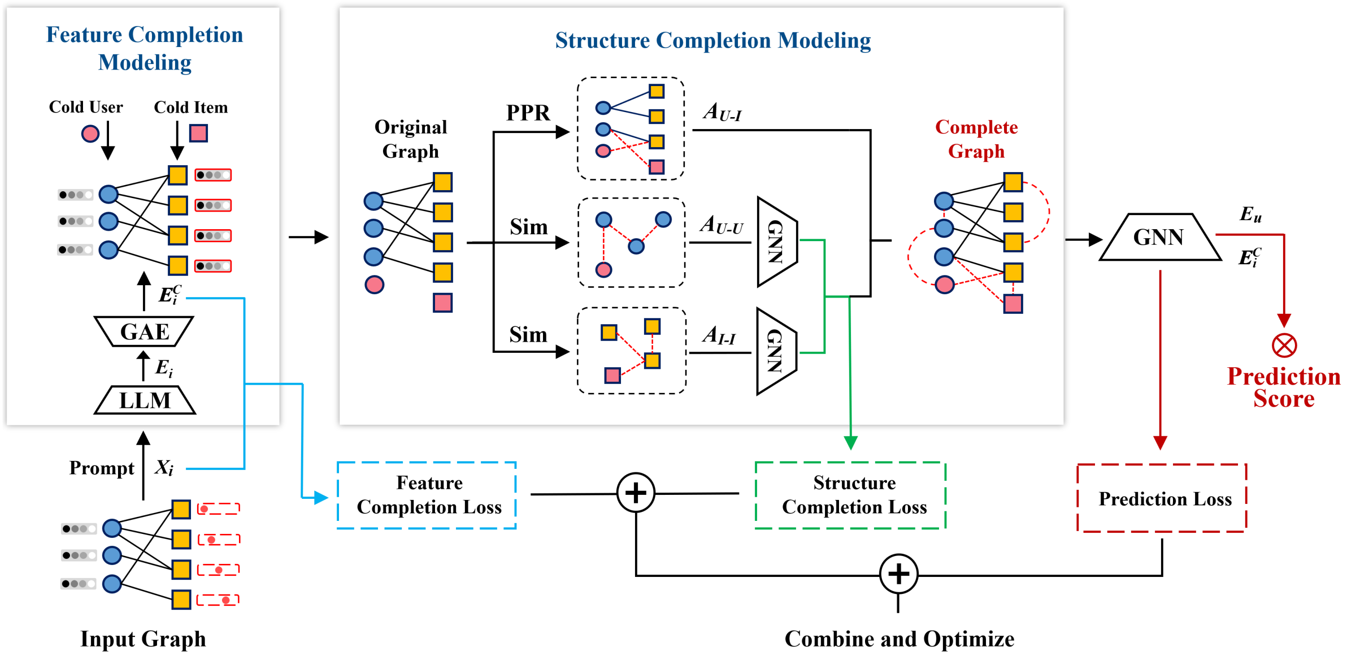


Figure 1: The architecture of FS-GNN.

methods to complete both features and structures to learn more meaningful and robust node embeddings, which are able to adapt to both warm and cold-start scenarios. The overall architecture of our proposed FS-GNN is shown in Fig. 1, which consists of three main components: (1) feature completion modeling, (2) structure completion modeling, as well as (3) model optimization. Specifically, we first apply LLMs to enhance original features and use GAE to denoise them, mitigating missing feature issues. For structure completion, we implement a three-channel model: the first channel uses the PPR algorithm to identify and add relevant user-item pairs, while the other two channels leverage feature similarity to connect nodes of the same type. Finally, we combine completion and prediction losses for end-to-end model optimization, yielding robust node embeddings suitable for score predictions in cold-start scenarios.

Feature Completion Modeling

In order to solve the problem that nodes of a certain class have no features, we divide the feature completion process into two steps. First, we utilize knowledgeable LLMs to generate the missing features. Then, these features are denoised and enhanced using GAE.

LLMs for Feature Completion. LLMs are trained on a large amount of real-world knowledge, enabling them to understand user preferences and provide valuable information (Wei et al. 2023). For items, features can be directly completed using their original features. For example, based on a movie's title, we can complete its director, country of release, and language used. For users, features can be completed through their historical interactions with items. For instance, by analyzing the features of the movies a user has watched, we can determine their preferred genres, favorite directors, and the country they are likely from (Wei et al. 2023). Specially, LLM-based feature completion consists of two steps:

(1) Construct prompts for each user/item for feature completion, and then input the prompt into LLM (e.g., gpt-3.5-turbo) to generate augmented features for each user/item.

$$\mathbf{X}'_u = \text{LLM}(\mathbf{P}(\mathbf{X}_u, \mathbf{A}_{U-I})), \quad \mathbf{X}'_i = \text{LLM}(\mathbf{P}(\mathbf{X}_i)) \quad (2)$$

where $\mathbf{P}(\cdot)$ represents the prompt construction function, and we complete user features through the users' historical interactions $\mathbf{A}_{U-I}$ and the initial textual features $\mathbf{X}_u$, while we complete item features directly using the initial textual features $\mathbf{X}_i$.

(2) Input the generated features into LLM with encoding capability (e.g., text-embedding-ada-002) to obtain augmented node embeddings. Then, we use principal components analysis (PCA) for dimensionality reduction to map these completed features into their respective spaces.

$$e_u = \text{PCA}(\text{Enc}(\mathbf{X}'_u)), \quad e_i = \text{PCA}(\text{Enc}(\mathbf{X}'_i)) \quad (3)$$

where e_u and e_i represent the final node embeddings for user u and item i. By fully utilizing original features and LLMs, user preferences are modeled from a natural language perspective, rather than relying solely on interactions.

Graph Auto-Encoder for Feature Augmentation. Features completed by LLMs are not always valid, and noise may affect the recommendation. To this end, we need to further reconstruct the features. Here, we take the scenario where user features are complete and item feature is missing (i.e., $u \in U^+, i \in I^-$) as an example, and note that our method is still useful for the scenarios where item features are complete and user features are missing. Inspired by Graph Attention Auto-Encoders (GATE) (Salehi and Davulcu 2020), we first use Graph Attention Networks (GAT) (Velickovic et al. 2018) as an encoder to evaluate the importance between user and item nodes using the attention mechanism (Lee, Lee, and Kang 2019), and determine which nodes in the direct-connected set U^+ are best suited for contributing features to the nodes with missing features in I^-. Then we rebuild and complete the features of the nodes in the set I^- through a decoder. The encoder and decoder are implemented as follows:

(1) **Encoder.** Given a direct-connected user u and item i, we compute the importance a_{ui} through the attention layer, which means the contribution of the node u to node i. Since the user and item nodes have different embedding spaces (Wang et al. 2019), we first transform them to the same embedding space and then compute the attention score:

$$\alpha_{iu} = \frac{\exp(\text{LeakyReLU}(z_{iu}))}{\sum_{k\in\mathcal{N}_i}\exp(\text{LeakyReLU}(z_{ik}))}, \quad \forall u \in \mathcal{N}_i, \quad (4)$$

where $z_{iu} = \mathbf{a}^T[\mathbf{W}\mathbf{e}_i \| \mathbf{W}\mathbf{e}_u]$, $\mathbf{W} \in \mathbb{R}^{D'\times D}$ and $\mathbf{a} \in \mathbb{R}^{2D'\times 1}$ denote linear transformation, T represents transposition, $\|$ is the concatenation operation, $\mathcal{N}_i$ denotes the neighbor set of node i, and a LeakyReLU with negative input slope $\beta = 0.2$ as activation function.

To mitigate the effects of noise and further make our model more suitable for cold-start scenarios, we aim to generate node embeddings that insensitive to models. Inspired by the Denoising Auto-Encoders (DAE) (Vincent et al. 2010), we first randomly drop out some features of the nodes in I^-, denoted as:

$$\tilde{\mathbf{e}}_i = \mathbf{e}_i \odot \mathbf{d}_i, \quad (5)$$

where $\odot$ denotes the Hadamard product, and $\mathbf{d}_i$ is a mask vector sampled independently from the Bernoulli distribution, where each element has a p_{drop} probability of 0, and a $1 - p_{drop}$ probability of 1. This forces the model to capture and retain more meaningful node embeddings during feature reconstruction.

Then, based on the attention coefficient α_{iu} aggregate information from the neighbors $\mathcal{N}_i$ to compute the embedding of the central node as follows:

$$\mathbf{e}_i^C = \text{sigmoid}\left(\sum_{u\in\mathcal{N}_i}\alpha_{iu}\mathbf{e}_u + \tilde{\mathbf{e}}_i\right). \quad (6)$$

Thus we obtain new node embeddings that aggregate the information of neighbor features.

(2) **Decoder.** The goal of the decoder is to reconstruct node embeddings obtained by the encoder. We use Graph Convolutional Network (GCN) (Kipf and Welling 2017) as the backbone of the decoder. Meanwhile, in order to make the reconstructed features as close as possible to the original features, we use the Mean Square Error (MSE) as a loss function to measure the quality of the reconstructed features and optimize the parameters of the feature completion modeling, that is:

$$\mathcal{L}_{\text{fc}} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{e}_i - \mathbf{e}_i^C)^2. \quad (7)$$

Structure Completion Modeling

For the user-item bipartite graph in the cold-start scenario, there is a lack of interactions between U-I and no connections between nodes of the same type, making it difficult to predict users' preferences. For this purpose, we design three channels to complete the structures between U-U, I-I, and U-I, respectively. Specifically, a Personalized PageRank (PPR) (Klicpera, Bojchevski, and Günnemann 2019) is used to complete the structure between U-I and we complete the structure between U-U and I-I by constructing a kNN graph (Monti, Bronstein, and Bresson 2017).

Completing the Relations between U-I. For completing the structure between different types of nodes, we use the PPR algorithm, which helps us to find items of interest to the user to extend the incomplete graph structure. Specifically, the PPR algorithm takes the user-item bipartite graph as input, and uses a restarted random wandering strategy to compute the probability $P_u = \{p_{u,i_1}, p_{u,i_2}, \ldots, p_{u,i_N}\}$ that user u visits each item node. A larger p_{ui} indicates a stronger correlation between two nodes. Formally, the PPR matrix $\mathbf{P}$ can be computed as:

$$\mathbf{P} = (1-\delta)\hat{\mathbf{A}}\mathbf{P} + \delta\mathbf{I}, \quad (8)$$

where δ is the reset probability, and $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$. Considering that there are inevitably some nodes in $\mathbf{P}$ with low visit probability, we keep only the top k nodes in $\mathbf{P}$ with high probability, so as to obtain the PPR-based structure matrix $\mathbf{A}_{U-I}$.

Completing the Relations between U-U and I-I. Interactions between users (e.g., friendships, common interests) are important to the recommendation process. Based on the homogeneity assumption (McPherson, Smith-Lovin, and Cook 2001) that similar individuals tend to be connected to each other, we infer that users with similar features (e.g., gender, age) are more likely to exhibit similar preferences, thus the similarity of features between two user nodes can be calculated by using cosine similarity (Qian et al. 2022):

$$S_{mn} = \frac{\mathbf{e}_u^{(m)} \cdot \mathbf{e}_u^{(n)}}{\|\mathbf{e}_u^{(m)}\|\|\mathbf{e}_u^{(n)}\|}, \quad (9)$$

and connect top k similar users:

$$\mathbf{A}_{U-U}(m,n) = \begin{cases} S_{mn} & \text{if } n \in \text{TopK}(S_{m\cdot}), \\ 0 & \text{otherwise.} \end{cases} \quad (10)$$

where S_{mn} represents the similarity between users m and n, and $S_{m\cdot}$ denotes the similarity vector of node m with all other nodes.

Similarly, relationships between items (e.g., co-browsed, co-purchased) can be used to discover similar or complementary items. Therefore, we also sample the top k nodes to connect based on similarity:

$$\mathbf{A}_{I-I}(m,n) = \begin{cases} S_{mn} & \text{if } n \in \text{TopK}(S_{m\cdot}), \\ 0 & \text{otherwise.} \end{cases} \quad (11)$$

Structure Joint Optimization. To ensure that the structure completion process is learnable, we propose a structure joint optimization strategy. Firstly, we feed the matrix $\mathbf{A}_{U-U}$ and the user embedding matrix $\mathbf{E}_u$ as an input into GCN (Kipf and Welling 2017), so as to make the user embeddings to be aggregated and propagated on the graph:

$$\mathbf{E}_u^{(l+1)} = \text{ReLU}\left(\tilde{\mathbf{D}}_u^{-\frac{1}{2}}\tilde{\mathbf{A}}_{U-U}\tilde{\mathbf{D}}_u^{-\frac{1}{2}}\mathbf{E}_u^{(l)}\mathbf{W}_u^{(l+1)}\right), \quad (12)$$

where $\mathbf{W}_u^{(l+1)}$ is the weight matrix of the $(l+1)$-th layer, $\tilde{\mathbf{A}}_{U-U} = \mathbf{A}_{U-U} + \mathbf{I}_u$, and $\tilde{\mathbf{D}}_u$ is the diagonal degree matrix of $\tilde{\mathbf{A}}_{U-U}$. In this way, the specific structure information of $\mathbf{A}_{U-U}$ can be obtained.

Similarly, we input the matrix $\mathbf{A}_{I-I}$ along with the feature-completed item embedding matrix $\mathbf{E}_i^C$ into another independent GCN to compute the specific item embeddings learned from $\mathbf{A}_{I-I}$ in the same way as on $\mathbf{A}_{U-U}$.

To further guide the process of structure completion and enhance the higher-order connectivity among the same type of nodes, we adopt a supervised optimization approach, which connects the final user and item embeddings and input them into an MLP to obtain the predicted values $\hat{y}_{ui}$:

$$\hat{y}_{ui} = \text{MLP}\left(\mathbf{e}_u \| \, \mathbf{e}_i^C\right). \quad (13)$$

We also construct an array y of all 1's as a label, which has the same dimensions as the predicted values, indicating that a connection exists between the corresponding node pairs. Then, the MSE is used to measure the final structure completion loss:

$$\mathcal{L}_{sc} = \frac{1}{|\mathcal{E}|}\sum_{\substack{(u,u)\in\mathcal{E} \\ (i,i)\in\mathcal{E}}} (y - \hat{y}_{ui})^2 \quad (14)$$

where $\mathcal{E}$ denotes the edges in $\mathbf{A}_{U-U}$ and $\mathbf{A}_{I-I}$.

Finally, considering the initial interactions carry rich and useful information, we merge $\mathbf{A}_{U-I}$ into the original interaction matrix $\mathbf{R}$ to get the final complete adjacency matrix:

$$\mathbf{A}^C = \begin{bmatrix} \mathbf{A}_{U-U} & \mathbf{R} \cup \mathbf{A}_{U-I} \\ (\mathbf{R} \cup \mathbf{A}_{U-I})^T & \mathbf{A}_{I-I} \end{bmatrix}. \quad (15)$$

Model Optimization

Model Prediction. After obtaining the complete graph and rich node embeddings, we learn the node embeddings over the complete graph with the help of a LightGCN (He et al. 2020). In this way, we can fully capture the total information from the feature completion module and structure completion module.

LightGCN employs a simple weighted sum aggregator, which removes heavy feature transformations and nonlinear activations. The graph convolutional operation is defined as:

$$\begin{aligned} \mathbf{e}_u^{(l+1)} &= \sum_{i\in\mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}}\mathbf{e}_i^{(l)}, \\ \mathbf{e}_i^{(l+1)} &= \sum_{u\in\mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_i|}\sqrt{|\mathcal{N}_u|}}\mathbf{e}_u^{(l)}. \end{aligned} \quad (16)$$

where $\frac{1}{\sqrt{|\mathcal{N}_u|}\sqrt{|\mathcal{N}_i|}}$ denotes the symmetric normalization term, avoiding the embedding size increasing with the graph convolutional operation. After k layers graph convolutional operations, we combine the embeddings obtained at each layer to form the final embedding of a user or an item:

$$\mathbf{e}_u = \sum_{k=0}^{K} \alpha_k \mathbf{e}_u^{(k)}, \quad \mathbf{e}_i = \sum_{k=0}^{K} \alpha_k \mathbf{e}_i^{(k)}. \quad (17)$$

In addition, we follow (He et al. 2020) and set α_k in the above equation to $1/(K+1)$ uniformly. By adopting layer combination, our model can be able to efficiently integrate information from neighborhoods with different hops.

Then, we fuse the embeddings of users and items to obtain a prediction score:

$$\hat{r}_{ui} = \text{sigmoid}\left(\text{MLP}\left(\mathbf{e}_u \| \mathbf{e}_i^C\right)\right). \quad (18)$$

Finally, we minimize the MSE loss to learn the user preferences:

$$\mathcal{L}_{pred} = \frac{1}{|\mathbf{R}|}\sum_{(u,i)\in\mathbf{R}} (r_{ui} - \hat{r}_{ui})^2, \quad (19)$$

where r_{ui} is the actual rating of user u and item i.

Overall Objective Function. We jointly optimize the feature completion task, the structure completion task and the recommendation task, and can obtain the following overall objective function:

$$\mathcal{L} = \lambda\mathcal{L}_{fc} + \mu\mathcal{L}_{sc} + \mathcal{L}_{pred}, \quad (20)$$

where λ and μ represent the weighted loss coefficient for the feature completion modeling and the structure completion modeling, respectively. Besides, we further introduce an adaptive tuning mechanism that dynamically adjusts these weight values based on feedback from the model's performance on specific subtasks, ensuring that our proposed FS-GNN is flexible enough to different cold-start scenarios.

Experiments

In this section, we conduct extensive experiments on three public recommendation datasets to validate the effectiveness of our proposed FS-GNN in various cold-start recommendation scenarios.

Experimental Setup

Datasets. We select three widely used public benchmark datasets, namely MovieLens100K, MovieLens1M[1], and Yelp[2], to verify the effectiveness of our proposed FS-GNN. MovieLens is a movie rating dataset where movies have only one feature genre, and user features include gender, age, and occupation. Yelp is a business rating dataset where users have only one feature, either an ID or name, and other features are unavailable due to privacy reasons. The features of the business include star rating, city, reviews, etc. Across these three datasets, user ratings for items range from 1 to 5.

For each dataset, we divide it into training set, validation set and testing set with 80%, 10% and 10% respectively.

[1]https://grouplens.org/datasets/movielens/

[2]https://www.yelp.com/dataset/challenge

MAE ↓	MovieLens100K			MovieLens1M			Yelp		
Methods	UCS	ICS	WS	UCS	ICS	WS	UCS	ICS	WS
NFM	0.8503	0.9043	0.8603	0.8271	0.8595	0.8397	0.8921	0.9482	0.8821
DropoutNet	0.8652	0.9323	0.8752	0.8432	0.9403	0.8647	0.8874	0.9756	0.9146
LightGCN	0.8909	0.9194	0.9031	0.9148	0.9221	0.9968	0.9627	0.9843	0.9632
MetaEmb	0.8249	0.8622	0.7886	0.8003	0.8762	0.7433	0.8849	0.8925	0.8274
MeLU	0.8834	0.9067	0.7821	0.8469	0.9199	0.7456	0.9006	0.9481	0.8388
MetaHIN	0.8576	0.8805	0.8156	0.8435	0.9044	0.8071	0.8981	0.9121	0.8723
ColdGPT	0.7802	0.8856	**0.7149**	0.8091	0.8601	0.7315	0.9005	0.8723	0.8254
MeGNN	0.7983	0.8572	0.7492	0.7957	0.8569	0.7379	0.8637	0.8693	0.8227
AGNN	0.8125	0.8576	0.7228	0.8012	0.8411	**0.6944**	0.8609	0.8826	**0.7997**
FS-GNN	**0.7735**	**0.8064**	0.7362	**0.7653**	**0.7671**	0.7103	**0.8492**	**0.8002**	0.8134

RMSE ↓	MovieLens100K			MovieLens1M			Yelp		
Methods	UCS	ICS	WS	UCS	ICS	WS	UCS	ICS	WS
NFM	1.0532	1.1331	1.0982	1.0298	1.1294	1.0732	1.1162	1.1349	1.1032
DropoutNet	1.0786	1.1401	1.1056	1.0482	1.1602	1.0897	1.1029	1.1831	1.1377
LightGCN	1.0712	1.1261	1.1046	1.0958	1.1046	1.1052	1.1544	1.1664	1.0902
MetaEmb	1.0465	1.0858	1.0032	0.9943	1.1078	0.9337	1.0923	1.1009	1.0243
MeLU	1.0734	1.1143	0.9828	1.0866	1.1331	0.9208	1.1215	1.1528	1.0487
MetaHIN	1.0581	1.0632	0.9563	1.0702	1.0976	0.9455	1.0969	1.1137	1.0805
ColdGPT	1.0134	1.0864	**0.9163**	1.0421	1.0792	0.9272	1.1209	1.0911	1.0263
MeGNN	0.9905	1.0583	0.9582	1.0101	1.0597	0.9374	1.0367	1.0806	1.0344
AGNN	1.0347	1.0575	0.9291	0.9906	1.0401	**0.8941**	1.0611	1.0809	**1.0175**
FS-GNN	**0.9635**	**1.0021**	0.9219	**0.9669**	**0.9678**	0.9063	**0.9928**	**1.0031**	1.0204

Table 1: Comparison experiments. The best results are shown in bold, and the second best results are underlined.

Baselines. In order to verify the validity of our model, we compare FS-GNN with three types of methods: (1) **Traditional Methods**, including NeuMF (He et al. 2017), DropoutNet (Volkovs, Yu, and Poutanen 2017) and LightGCN (He et al. 2020). (2) **Meta-learning-based Methods**, including MetaEmb (Pan et al. 2019), MeLU (Lee et al. 2019) and MetaHIN (Lu, Fang, and Shi 2020). (3) **GNN-based Methods**, including MeGNN (Liu et al. 2023), ColdGPT (Cao et al. 2023) and AGNN (Qian et al. 2022).

Implementation Details. For our proposed FS-GNN, for feature completion module, when we use LLM to complete the user features, we utilize up to 30 historical interactions to ensure that the token does not exceed the limit. We use GAT as encoder and GCN as decoder with $\{64, 128\}$ units in the hidden layer, and set the p_{drop} to 0.3. For structure completion module, we set the top k for both kNN-based and PPR-based structure completion strategies in $\{10, 15, \ldots, 25\}$. We set the weighted loss coefficient λ for the feature completion module to 0.5, and the weighted loss coefficient μ for the structure completion module to 0.5, the learning rate to 0.005 and weight decay to 0.0005. We set the embedding dimension to 64 for all compared methods. We use Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) as evaluation metrics, both of which are widely used in rating prediction tasks, such as (Qian et al. 2022; Zheng et al. 2021). For LightGCN, which serves as the base model of our FS-GNN, we use its original parameter settings in the paper.

Performance Comparison

We compare FS-GNN with state-of-the-art baselines in two cold-start scenarios (UCS, ICS) and a traditional warm-start scenario (WS). The results are shown in Table 1.

Cold-start Scenarios. As shown in the first two parts of Table 1, we can find that FS-GNN exhibits the best performance among all compared models. Particularly, in the UCS scenario, our proposed FS-GNN improves over the state-of-the-art baseline AGNN by 6.9%, 5.5%, and 6.4% on the MovieLens100K, MovieLens1M, and Yelp datasets, respectively. Similarly, in the ICS scenario, FS-GNN outperforms AGNN by 6.0%, 8.8%, and 9.3% on the same datasets. This proves that the complementary feature information and structure information can effectively solve the cold-start problem. To be specific, FS-GNN significantly outperforms the traditional recommend methods, i.e., NFM and LightGCN, by 9.3% and 13.6% on average, since they cannot fully utilize the feature information of users and items. Compared with methods that are based on Meta-learning, such as MetaEmb, MeLU, MetaHIN, our FS-GNN achieves an improvement between 6.2% and 12.4% in terms of mean accuracy, showing that our method also performs well when data is scarce. In addition, GNN-based approaches have shown good performance in cold-start scenarios, especially AGNN, which manually introduces external knowledge to enrich the data information, which requires a significant cost. In contrast, FS-GNN can automatically complete the features with rich knowledge base of LLMs. These results demonstrate that the model we designed can maximally exploit the information in the original data, obtaining more robust node representations to alleviate the cold-start problem.

Warm-start Scenario. As shown in the last part of Table 1, we can find that our proposed FS-GNN achieves sub-optimal performance compared to all baselines, slightly lower than AGNN and ColdGPT. This demonstrates that our proposed feature-structure adaptive completion network effectively addresses data sparsity in recommendation scenarios. This also highlights the potential of graph completion methods in recommender systems.

Ablation Study

To further investigate the contribution of each component of our proposed FS-GNN to the final recommendation performance, we design three variants: (1) **LightGCN** (He et al. 2020): which serves as the base model of our FS-GNN. (2) **FC-GNN**: a variant of FS-GNN which retains only the feature completion modeling based on graph auto-encoder, that is, it preserves only the feature completion loss $\mathcal{L}_{fc}$. (3) **SC-GNN**: a variant of FS-GNN which retains only the three-channel structure completion modeling, meaning it keeps only the structure completion loss $\mathcal{L}_{sc}$.

From the results in Table 2, we can draw the following conclusions: (1) FS-GNN consistently outperforms all variants, indicating the effectiveness of using both feature and structure completion modelings. (2) Compared with LightGCN, the results of FC-GNN and SC-GNN are improved by 6.1% and 13.8%, respectively, verifying the usefulness of the two components. This significant improvement highlights the importance of completing the feature and structure information. (3) SC-GNN outperforms FC-GNN in both cold-start and warm-start scenarios, which implies that the structure completion modeling makes a significant contribution to our model.

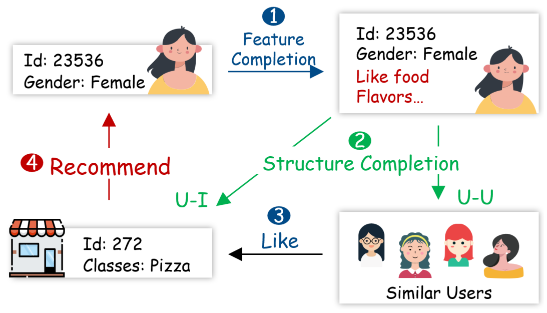
Scenarios	Methods	MovieLens100K		MovieLens1M		Yelp	
		MAE	RMSE	MAE	RMSE	MAE	RMSE
UCS	LightGCN	0.8909	1.0712	0.9148	1.0958	0.9627	1.1544
	FC-GNN	0.8359	1.0213	0.8427	1.0551	0.9305	1.1132
	SC-GNN	0.7895	0.9842	0.7887	0.9901	0.8531	0.9936
	FS-GNN	**0.7735**	**0.9635**	**0.7653**	**0.9669**	**0.8492**	**0.9828**
ICS	LightGCN	0.9194	1.1261	0.9221	1.1046	0.9843	1.1664
	FC-GNN	0.8805	1.0923	0.8711	1.0872	0.8967	1.0952
	SC-GNN	0.8804	1.0735	0.8731	1.0627	0.8893	1.0897
	FS-GNN	**0.8064**	**1.0021**	**0.7671**	**0.9678**	**0.8002**	**1.0031**
WS	LightGCN	0.9031	1.1046	0.9968	1.1052	0.9632	1.0902
	FC-GNN	0.8485	1.0482	0.8347	1.0432	0.9023	1.1033
	SC-GNN	0.7457	0.9421	0.7313	0.9195	0.8395	1.0429
	FS-GNN	**0.7362**	**0.9319**	**0.7203**	**0.9103**	**0.8134**	**1.0204**

Table 2: Performance comparison of FS-GNN variants.

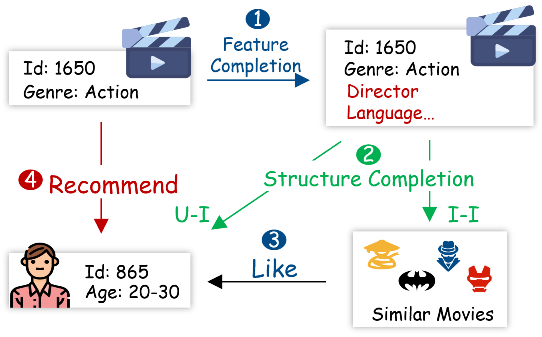
Case Study

In addition to effective performance, FS-GNN is also well interpretable. We illustrate its capacity to address the cold-start problem using two scenarios from Yelp and MovieLens.

Case 1 (Existing Items for New Users). As shown in Fig. 2a, a newly registered Yelp user (ID: 23536) fills minimal



(a) Existing items for new users



(b) New items for existing users

Figure 2: Case Study for New user and New item.

personal info and hasn't rated any merchants. FS-GNN employs the feature completion modeling to enhance her profile. Subsequently, the structure completion identifies communities and users who share same interests. Finally, recommendations of merchants that are of interest to similar users are presented to her.

Case 2 (New Items for Existing Users). As shown in Fig. 2b, for a new movie (ID: 1650) that only has a title and genre, additional information like the director and language is missing. FS-GNN enriches its features through the feature completion modeling and obtain more informative embeddings by identifying similar movies through the structure completion modeling. In this way, the movie is recommended to users who are interested in similar movies.

Conclusion

In this paper, we address the cold-start problem by proposing a new model called FS-GNN. Through a trainable feature completion module and a tri-channel structure completion module, FS-GNN enhances node embeddings and completes user-user, item-item, and user-item structures. We adaptively integrate these modules in an end-to-end fashion, creating more comprehensive and robust embeddings for recommendation tasks. Experimental results on multiple public datasets demonstrate significant improvements of our proposed FS-GNN in cold-start scenarios. In the future, we plan to explore more graph completion methods and consider how to incorporate ideas from adversarial or contrastive learning to further address the cold-start problem.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 92370111, No. 62422210, No. 62272340, No. 62276187 and No. 62402337), the Postdoctoral Fellowship Program of CPSF under Grant No. GZC20241207, and the China Postdoctoral Science Foundation under Grant No. 2024M752367.

References

Cao, Y.; Yang, L.; Wang, C.; Liu, Z.; Peng, H.; You, C.; and Yu, P. S. 2023. Multi-task Item-attribute Graph Pre-training for Strict Cold-start Item Recommendation. *Proceedings of the 17th ACM Conference on Recommender Systems*.

Chen, X.; Chen, S.; Yao, J.; Zheng, H.; Zhang, Y.; and Tsang, I. W. 2022. Learning on Attribute-Missing Graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(2): 740–757.

Covington, P.; Adams, J. K.; and Sargin, E. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*,

Dong, M.; Yuan, F.; Yao, L.; Xu, X.; and Zhu, L. 2020. MAMO: Memory-Augmented Meta-Optimization for Cold-start Recommendation. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*,

Dong, Y.; He, D.; Wang, X.; Li, Y.; Su, X.; and Jin, D. 2023. A Generalized Deep Markov Random Fields Framework for Fake News Detection.

Goldberg, D.; Nichols, D. A.; Oki, B. M.; and Terry, D. B. 1992. Using Collaborative Filtering to Weave an Information Tapestry. *Commun. ACM*, 35(12): 61–70.

He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; and Wang, M. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*,

He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; and Chua, T.-S. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web*,

Huang, F.; Yang, Z.; Jiang, J.; Bei, Y.-Q.; Zhang, Y.; and Chen, H. 2024. Large Language Model Interaction Simulator for Cold-Start Item Recommendation. *ArXiv*, abs/2402.09176.

Jin, D.; Huo, C.; Liang, C.; and Yang, L. 2021. Heterogeneous Graph Neural Network via Attribute Completion. In *Proceedings of the Web Conference*,

Kipf, T.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks.

Klicpera, J.; Bojchevski, A.; and Günnemann, S. 2019. Predict then Propagate: Graph Neural Networks meet Personalized PageRank.

Koren, Y.; Bell, R. M.; and Volinsky, C. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer*, 42(8): 30–37.

Lee, H.; Im, J.; Jang, S.; Cho, H.; and Chung, S. 2019. MeLU: Meta-Learned User Preference Estimator for Cold-Start Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on nowledge Discovery & Data Mining*,

Lee, J.; Lee, I.; and Kang, J. 2019. Self-Attention Graph Pooling. In *Proceedings of the 36th International Conference on Machine Learning*,

Liu, H.; Lin, H.; Zhang, X.; Ma, F.; Chen, H.; Wang, L.; Yu, H.; and Zhang, X. 2023. Boosting Meta-Learning Cold-Start Recommendation with Graph Neural Network. *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*.

Lu, Y.; Fang, Y.; and Shi, C. 2020. Meta-learning on Heterogeneous Information Networks for Cold-start Recommendation. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*,

McPherson, M.; Smith-Lovin, L.; and Cook, J. M. 2001. Birds of a Feather: Homophily in Social Networks. *Review of Sociology*, 27: 415–444.

Monti, F.; Bronstein, M. M.; and Bresson, X. 2017. Geometric Matrix Completion with Recurrent Multi-Graph Neural Networks. In *Proceedings of Advances in Neural Information Processing Systems*,

Pan, F.; Li, S.; Ao, X.; Tang, P.; and He, Q. 2019. Warm Up Cold-start Advertisements: Improving CTR Predictions via Learning to Learn ID Embeddings. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, ACM.

Qian, T.; Liang, Y.; Li, Q.; and Xiong, H. 2022. Attribute Graph Neural Networks for Strict Cold Start Recommendation. *IEEE Trans. Knowl. Data Eng.*, 34(8): 3597–3610.

Salehi, A.; and Davulcu, H. 2020. Graph Attention Auto-Encoders. In *32nd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2020, Baltimore, MD, USA, November 9-11, 2020*, IEEE.

Salha-Galvan, G.; Hennequin, R.; Chapus, B.; Tran, V.-A.; and Vazirgiannis, M. 2021. Cold Start Similar Artists Ranking with Gravity-Inspired Graph Autoencoders. *Proceedings of the 15th ACM Conference on Recommender Systems*.

Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio', P.; and Bengio, Y. 2018. Graph Attention Networks.

Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; and Manzagol, P.-A. 2010. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *J. Mach. Learn. Res.*, 11: 3371–3408.

Volkovs, M.; Yu, G.; and Poutanen, T. 2017. DropoutNet: Addressing Cold Start in Recommender Systems. In *Proceedings of Advances in Neural Information Processing Systems*,

Wang, S.; Zhang, K.; Wu, L.; Ma, H.; Hong, R.; and Wang, M. 2021. Privileged Graph Distillation for Cold Start Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*,

Wang, X.; Dong, Y.; Jin, D.; Li, Y.; Wang, L.; and Dang, J. 2023. Augmenting Affective Dependency Graph via Iterative Incongruity Graph Learning for Sarcasm Detection.

Wang, X.; Ji, H.; Shi, C.; Wang, B.; Cui, P.; Yu, P. S.; and Ye, Y. 2019. Heterogeneous Graph Attention Network. In *Proceedings of the Web Conference*,

Wang, Z.; Mu, C.; Hu, S.; Chu, C.; and Li, X. 2022. Modelling the Dynamics of Regret Minimization in Large Agent Populations: a Master Equation Approach.

Wei, W.; Ren, X.; Tang, J.; Wang, Q.; Su, L.; Cheng, S.; Wang, J.; Yin, D.; and Huang, C. 2023. LLMRec: Large Language Models with Graph Augmentation for Recommendation. *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*.

Yang, L.; Zhou, W.; Peng, W.; Niu, B.; Gu, J.; Wang, C.; Cao, X.; and He, D. 2022. Graph Neural Networks Beyond Compromise Between Attribute and Topology. In *Proceedings of the Web Conference*,

Ying, R.; He, R.; Chen, K.; Eksombatchai, P.; Hamilton, W. L.; and Leskovec, J. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*,

Zheng, J.; Ma, Q.; Gu, H.; and Zheng, Z. 2021. Multi-view Denoising Graph Auto-Encoders on Heterogeneous Information Networks for Cold-start Recommendation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*,

Zhu, Y.; Lin, J.; He, S.; Wang, B.; Guan, Z.; Liu, H.; and Cai, D. 2020. Addressing the Item Cold-Start Problem by Attribute-Driven Active Learning. *IEEE Trans. Knowl. Data Eng.*, 32(4): 631–644.