

Graph Neural Patching for Cold-Start Recommendations

Hao Chen¹, Yu Yang², Yuanchen Bei³, Zefan Wang⁴,
Yue Xu⁵, and Feiran Huang⁴

¹ The Hong Kong Polytechnic University, Hong Kong SAR, China

² The Education University of Hong Kong, Hong Kong SAR, China

³ Zhejiang University, Hangzhou, China

⁴ Jinan University, Guangzhou, China

⁵ Alibaba Group, Hangzhou, China

sundaychenhao@gmail.com, yanggy@eduhk.hk

Abstract. The cold start problem in recommender systems remains a critical challenge. Current solutions often train hybrid models on auxiliary data for both cold and warm users/items, potentially degrading the experience for the latter. This drawback limits their viability in practical scenarios where the satisfaction of existing warm users/items is paramount. Although graph neural networks (GNNs) excel at warm recommendations by effective collaborative signal modeling, they haven't been effectively leveraged for the cold-start issue within a user-item graph, which is largely due to the lack of initial connections for cold user/item entities. Addressing this requires a GNN adept at cold-start recommendations without sacrificing performance for existing ones. To this end, we introduce Graph Neural Patching for Cold-Start Recommendations (GNP), a customized GNN framework with dual functionalities: GWarmer for modeling collaborative signal on existing warm users/items and Patching Networks for simulating and enhancing GWarmer's performance on cold-start recommendations. Extensive experiments on three benchmark datasets confirm GNP's superiority in recommending both warm and cold users/items.

Keywords: Cold-start recommendation · Graph neural networks · Collaborative filtering.

1 Introduction

Recommender systems aim to retrieve deserving items for users out of massive online content and provide personalized recommendations [2,8,10]. With countless new user registrations and item introductions occurring every second, these systems are frequently confronted with cold-start challenges, which is how to make recommendations for new users and items without historical interactions [4,11,26]. In a typical cold-start scenario for user-item recommendations, the system can access the auxiliary information for both warm and cold users/items to alleviate the cold-start problem [12,17,28].

A primary challenge in cold-start recommendations is to effectively cater to new users and items while ensuring that the recommendation quality for existing users and items remains unaffected, given the significant presence of both in the system. Given pre-trained warm embeddings and the auxiliary information, representative state-of-the-art models like DropoutNet [24] and Heater [32] cultivate hybrid models to recommend for both cold users/items and warm users/items. These hybrid models are tasked with handling two distinctly different types of inputs concurrently: (i) For warm recommendations, they utilize *warm embeddings* and the auxiliary information to craft warm representations. (ii) For cold-start recommendations, they rely on *placeholder embeddings* and the auxiliary information to produce a representation for new entities. While these hybrid cold-start models exhibit some proficiency in handling new entities, they often come at the cost of diminished performance for existing warm users and items. This trade-off occurs because the models must contend with handling two distinctly different types of inputs: the generic placeholder embeddings for newcomers and the pre-trained warm embeddings for the established base. The hybrid models struggle to deliver satisfactory recommendations for established users and items, which in turn limits the applicability of these solutions in real-world recommender systems.

Central to the effectiveness of a warm recommender system is the capability to accurately model the interactions among existing users and items. In this regard, Graph Neural Networks (GNNs), renowned for their adeptness in handling graph-structured data, have secured a leading position in numerous recommendation scenarios with effective collaborative filtering [10,25,31]. However, the direct application of current GNN models to address the cold-start problem is impeded by three primary challenges:

- Cold-start users and items lack historical interactions, which implies an absence of neighboring nodes within the graph structure. GNN models are unable to perform neighborhood aggregations, which are essential for refining the embedding representations of cold-start users and items, as they lack these neighboring connections.
- Obtaining precise representations for these new entities is inherently challenging. Secondly, crafting a GNN that can sustain recommendation performance for existing users and items while also effectively handling cold-start recommendations is far from a trivial task.
- Because of their recursive message passing mechanism, GNN models typically require more computational time compared to straightforward inner-product functions. Consequently, deploying GNN models for cold-start recommendations might face significant computational challenges, potentially outweighing the efficiency of current inner-product-based cold-start models.

To surmount the aforementioned challenges, this paper introduces a novel Graph Neural Patching model (GNP) designed to deliver satisfactory recommendations for both existing and cold-start users and items. The GNP framework consists of two distinct yet interconnected models: GWarmer and the Patching Network. GWarmer is a specially crafted, efficient GNN model that enhances

recommendations for existing users and items. Notably, GWarmer maintains a simplicity akin to inner-product calculations during inference, rendering the GNP framework more efficient than state-of-the-art cold-start models. Additionally, the Patching Network, which acts as a complementary module to GWarmer, provides cold-start recommendations by generating embeddings compatible with GWarmer from auxiliary information. In contrast to conventional cold-start models, the Patching Network is dedicated solely to cold-start recommendations, thereby ensuring that they do not disrupt the performance of the warm recommendation models. Extensive experiments on three benchmark datasets, with three representative types of embedding methods, present that GNP statistically significantly outperforms other recommendation models on both hybrid recommendation performance (including both warm and cold recommendation targets) and pure-warm recommendation performance with better efficiency. In summary, our contributions are listed as below:

- This paper pioneers the integration of cold-start recommendations within GNN models, culminating in enhanced recommendation performance across the board for both warm and newly encountered cold users and items.
- This paper addresses the trade-offs inherent in hybrid cold-start models and introduces Graph Neural Patching (GNP), a solution that effectively delivers cold-start recommendations without detracting from the performance of recommendations for existing warm users and items.
- Extensive experiments on three benchmark datasets, with three different embedding models, show that GNP significantly outperforms other cold-start models on all situations with statistically significant margins.

2 Related Works

2.1 Graph Neural Networks for Recommendations

Graph Neural Networks (GNNs) have become increasingly prominent in many relations modeling tasks [1,9,15,27]. For the domain of recommender systems, GNNs provide an effective framework for collaborative filtering to model the intricate user-item relationships [2,23,29]. GNNs for collaborative filtering operate by constructing a graph where nodes represent users and items, and edges represent interactions such as ratings or views [5,30]. The power of GNNs in this context stems from their capability to aggregate and transform node features through neural message passing, capturing both local and global dependencies within the graph. This allows the model to learn nuanced user and item embeddings that reflect their latent factors, which are critical for accurate recommendations. Representatively, NGCF [25] and LightGCN [10] construct bipartite graphs from historical user-item interactions and then embed the users and items with the help of graph convolutional networks [9,15].

2.2 Cold-Start Recommendations

The cold-start problem is a quintessential challenge in recommender systems, where the system must make personalized suggestions for new (cold) users or

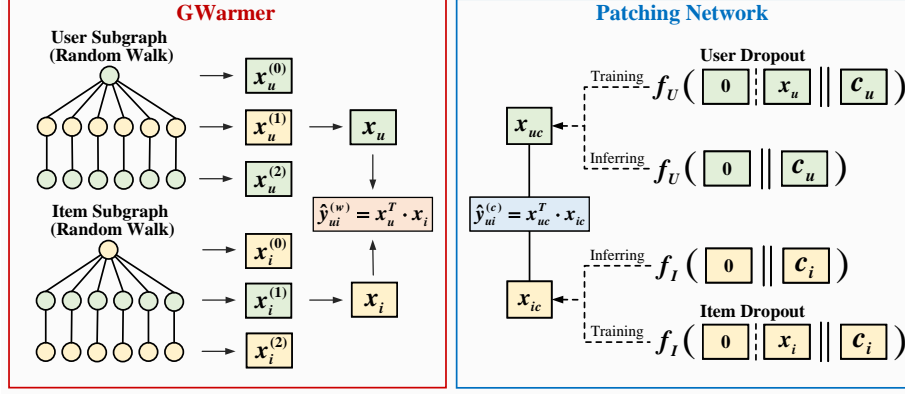


Fig. 1. The overall architecture of our proposed GNP. GWarmer provides recommendation for existing warm users and items, while the Patching Network flexibly recommend for cold-start users and items.

items that lack historical interaction data [4,12]. Traditional recommendation approaches, which rely heavily on past behavior, struggle with these new entities [21,22]. To tackle this issue, various strategies have been developed, such as utilizing auxiliary data like user demographics or item attributes, employing content-based filtering, and leveraging recommender models to predict preferences based on limited information. However, existing models mainly adopt separate modules for cold and warm recommendations respectively [24,32]. In this way, their effectiveness is frequently offset by a decline in recommendation performance for the warm users and items.

3 Methodology

This section first presents the whole framework of our Graph Neural Patching (GNP) model, and then we introduce the details of the proposed GWarmer and the Patching Network.

3.1 Overall Framework

Let $u \in \mathcal{U}$ denote the user index, and $i \in \mathcal{I}$ denote the item index. \mathcal{G} denotes the user-item bipartite graph constructed from the user-item interactions \mathcal{R} . \mathcal{U}_w and \mathcal{I}_w denote the existing (warm) user/item set. \mathbf{C}_U and \mathbf{C}_I denote the auxiliary information for all users/items. \mathbf{E}_U and \mathbf{E}_I denote the user/item embedding matrix trained by any given user-item embedding model.

We first define the general form of the warm/cold-start recommendation task. *Warm recommendations*, by definition, mean that recommend warm items to warm users. For warm users and items, we can access the warm embeddings

\mathbf{E}_U and \mathbf{E}_I and the historical interaction graph \mathcal{G} . Formally, GWarmer predicts the relevance of a warm user-item pair (u, i) with $\mathcal{W}(u, i; \mathcal{G}, \mathbf{E}_U, \mathbf{E}_I)$. By contrast, *Cold-start Recommendations* refer to the recommendation for either cold-start user or cold-start item since the recommendation model cannot access the historical interaction graph and the embeddings for both users and items simultaneously. Without the graph and the embeddings, Patching Networks generate their cold-start embeddings from their auxiliary information \mathbf{c}_u and \mathbf{c}_i , namely, $\mathcal{P}(u, i; \mathbf{c}_u, \mathbf{c}_i)$. After defining GWarmer and Patching Networks, for an arbitrary user-item pair, GNP computes its relevance scores with the following formula,

$$\hat{y}_{ui} = \begin{cases} \mathcal{W}(u, i; \mathcal{G}, \mathbf{E}_U, \mathbf{E}_I), & u \in \mathcal{U}_w \text{ and } i \in \mathcal{I}_w, \\ \mathcal{P}(u, i; \mathbf{c}_u, \mathbf{c}_i), & u \notin \mathcal{U}_w \text{ or } i \notin \mathcal{I}_w, \end{cases} \quad (1)$$

where $\mathcal{W}(\cdot)$ denotes GWarmer, and $\mathcal{P}(\cdot)$ denotes Patching Networks. We will detail these two parts in the next two subsections.

3.2 GWarmer

The prevalent models [10,13,20,21], predominantly concentrate on developing more sophisticated user and item embeddings. These embeddings are subsequently utilized to forecast relevance scores through an inner product operation, as delineated below:

$$\hat{y}_{ui}^{(w)} = \mathbf{e}_u^\top \mathbf{e}_i. \quad (2)$$

In alignment with this, an exemplary warm recommendation model tailored for practical recommender systems should possess the capability to refine any existing warm user-item embeddings by integrating them with the graph structure \mathcal{G} . This enhancement should be achievable without the need to dictate the specific embedding models used, encompassing a range of methods such as Matrix Factorization (MF) [16], MetaPath2Vec (M2V) [6], and LightGCN [10]. Furthermore, the model should be characterized by its efficiency in both the training phase and the inference process, ensuring that it meets the demands of real-time recommendation systems.

As depicted in Figure 1, the GWarmer model initiates its operation by constructing K -layer subgraphs for both warm users and items, leveraging the random-walk technique. This method effectively captures the intricate connections within the graph. Subsequently, the model aggregates these random-walk sets, thereby encapsulating the topological essence of the warm users and items within the graph structure. In the final stage, a self-adaptive inner product relevance function is designed to perform warm recommendations.

Random-Walk Step. The random-walk technique stands out as an efficient approach for capturing the essence of graph topology [7,19]. As delineated in Figure 1, for any designated warm user or item t , we systematically initiate S random walks, each spanning a length of K , with t established as the origin node. Here, S signifies the extent of our sampling endeavor. We collectively refer to this set of random walks as $\mathcal{T}_K(t)$. Note that in the context of bipartite user-item graphs, user nodes are interconnected with item nodes, and vice versa. To

provide a clear visual distinction in Figure 1, we employ green circles to represent user nodes and yellow circles to represent item nodes.

Walk Pooling Step. Upon acquiring the set of random walks, we proceed by replacing the node identifiers with their respective embedding vectors. Subsequently, we employ a mean pooling aggregator [3,9], to consolidate the random-walk set. Mean pooling is permutation-invariant on set elements, such that the order of the walks will not influence the aggregation. After the mean pooling aggregation, for node t we get its layer-wise representation for each layer of the subgraph. Formally, the layer-wise representation is computed using the following equation:

$$\mathbf{x}_t^{(k)} = \frac{1}{S} \sum_{\mathbf{T}_i \in \mathcal{T}_K(v)} \mathbf{T}_i^{(k)}, \quad (3)$$

where k denotes the layer number of the subgraph and S denotes the sampling size of the random walk. Specifically, $\mathbf{x}_t^{(0)} = E_t$.

Self-adaptive Inner Product. Through the aforementioned two steps, we have successfully generated K distinct representation vectors for each warm user or item t . Taking user u as an example, $\mathbf{x}_u^{(0)}$ is her own embeddings, while $\mathbf{x}_u^{(1)}$ contains the mean embedding of her interacted items, and $\mathbf{x}_u^{(2)}$ aggregates the embeddings of the users that have shared interacted items with u . This multi-layered approach provides a comprehensive profile of user u from various perspectives. Inspired by this multidimensional representation, we introduce a self-adaptive weighted sum mechanism that assigns unique weights to each layer. This further enhances the flexibility and extensibility of the inner product. By tailoring the user and item representations with distinct weights, the relevance function is articulated as follows:

$$\mathbf{x}_u = \sum_k w_u^{(k)} \mathbf{x}_u^{(k)}, \quad \mathbf{x}_i = \sum_k w_i^{(k)} \mathbf{x}_i^{(k)}, \quad (4)$$

$$\hat{y}_{ui}^{(w)} = \mathcal{W}(u, i; \mathcal{G}, \mathbf{E}_U, \mathbf{E}_I) = \mathbf{x}_u^\top \mathbf{x}_i, \quad (5)$$

where $\{w_u^{(0)}, \dots, w_u^{(K)}\}$ and $\{w_i^{(0)}, \dots, w_i^{(K)}\}$ denotes the self-adaptive weights for users' and items' layer-wise representations, respectively. \mathbf{x}_u and \mathbf{x}_i denotes the GWarmer representation of u and i .

In contrast to other recursive message passing models, GWarmer emerges as an exceptionally suitable candidate for large-scale recommender systems. The reasons for its suitability are manifold: Firstly, the random walk step and the walk pooling step are devoid of trainable parameters, which allows for their offline computation and storage in a parallelized fashion. Secondly, in large commercial systems where historical user-item interactions are typically housed in distributed databases, GWarmer's approach offers a distinct advantage. Unlike traditional recursive GNNs that demand considerable time for recursively expanding neighbors, GWarmer's random walk policy substantially reduces the I/O overhead. Finally, once the GWarmer representations are stored, the model's operation during the online inference process is as straightforward as computing

inner product functions. This simplicity makes GWarmer highly compatible with existing online recommender frameworks, such as Faiss [14].

3.3 Patching Network

Serving as a novel patch of GWarmer, the Patching Network needs to be consistent with GWarmer when conducting cold-start recommendation tasks. To ensure seamless integration, as depicted in Figure 1, we facilitate the connection between GWarmer and the Patching Network by leveraging GWarmer’s representations in lieu of raw embeddings for the training of the Patching Network. As shown in Figure 1, we employ the popular cold-start dropout mechanism [24,32] to optimize the Patching Networks by randomly masking the GWarmer representations of warm users/items with ratio τ . For any given warm node t , we generate a random variable p , which follows a Bernoulli distribution with parameter τ . The resulting masked GWarmer representation is articulated by the equation:

$$\mathcal{D}(\mathbf{x}_t, \tau) = p \cdot \mathbf{0} + (1 - p) \cdot \mathbf{x}_t, \quad (6)$$

where $\mathbf{0}$ is a zero vector that has the same dimension as \mathbf{x}_t . Subsequently, for a given warm user-item pair (u, i) , the Patching Network will predict its relevance scores using the following formula:

$$\mathbf{x}_{uc} = f_U(\mathcal{D}(\mathbf{x}_u, \tau) \parallel \mathbf{c}_u), \quad \mathbf{x}_{ic} = f_I(\mathcal{D}(\mathbf{x}_i, \tau) \parallel \mathbf{c}_i), \quad (7)$$

$$\hat{y}_{ui}^{(c)} = \mathcal{P}(u, i; \mathbf{c}_u, \mathbf{c}_i) = \mathbf{x}_{uc}^\top \mathbf{x}_{ic}, \quad (8)$$

where f_U and f_I denote the mapping function for users and items, respectively. Note that, p is set as 1 during the inference of cold-start recommendation tasks when we set the embeddings of cold-start users/items as the placeholder embedding $\mathbf{0}$.

Objective Function. In the GNP framework, the self-adaptive weights in Eq. (4) and the mapping functions in Eq. (7) require optimization. Thus, we employ the Mean Squared Error (MSE) loss [24,32] to optimize both GWarmer and the Patching Network simultaneously. Formally, the loss function contains two parts and we present it as:

$$\mathcal{L} = \sum_{\mathcal{R}^+ \cup \mathcal{R}^-} (y_{ui} - \mathcal{W}(u, i; \mathcal{G}, \mathbf{E}_U, \mathbf{E}_I))^2 + (y_{ui} - \mathcal{P}(u, i; \mathbf{c}_u, \mathbf{c}_i))^2, \quad (9)$$

where \mathcal{R}^+ denotes the observed user-item interactions, while \mathcal{R}^- denotes the sampling negative interactions. Thus, $y_{ui} = 1$ for $(u, i) \in \mathcal{R}^+$ and $y_{ui} = 0$ for $(u, i) \in \mathcal{R}^-$.

4 Experiments

In this section, we aim to answer the following two research questions: **RQ1**: How does GNP perform compared with state-of-the-art cold-start recommendation models? **RQ2**: What is the efficiency of the GNP framework? **RQ3**: How do the hyper-parameters, such as the dropout ratio τ , affect GNP’s performance?

4.1 Experimental Settings

Dataset Description. We conduct our experiments on three cold-start recommendation datasets (CiteULike, XING, and WeChat). CiteULike [32] is a user-article dataset, where each article has a 300-dimension TF-IDF vector. XING [32] is a user-view-job dataset where each job is described by a 2738-dimension vector. WeChat is a user-video dataset, where each video can be represented by its NLP descriptions with a 200-dimension vector. We summarize the statistics of these three datasets in Table 1. Here we mainly study the cold-start problem of items. For CiteULike and XING, 20% of items are selected to be the cold-start items. Excluding the cold-start items, 65% of historical interactions are used for embedding training, while 15% of historical interactions are used for training GNP, and the remaining 20% interactions are used for validation/testing. For WeChat, we split the dataset according to its timeline, where we use the newly produced videos as the cold-start items. We adopt the popular all-ranking evaluation protocol [10,25,32]. Specifically, we report Precision@K, Recall@K, and NDCG@K metrics, where K=20 for CiteULike and XING and K=100 for WeChat.

Table 1. Statistics of the three experimental datasets.

Dataset	# User	# Item	# Interaction	Density
CiteULike	5,551	16,980	204,986	0.22%
XING	106,881	20,519	3,856,580	0.18%
WeChat	87,772	446,408	17,768,997	0.05%

Baselines & Embedding Methods. We consider two SOTA cold-start recommendation models: DropoutNet [24] and Heater [32] as our baselines. To evaluate the universality of GNP, we choose three different types of embedding methods:

- MF [21] optimizes the matrix factorization with a pairwise ranking loss for recommendations, which is tailored to learn from implicit feedback.
- M2V [6] adopts metapath-based random walks on user-item graphs as corpus and then leverages skip-gram [18] models to compute node embeddings.
- LightGCN [10] utilizes GCN layers to aggregate neighbors’ information, which presents to be currently the most powerful embedding model for user-item recommendation.

Implementation. The embedding size is fixed to 200 for all models, and all the embedding methods are implemented with the official codes. We skip LightGCN embeddings on the WeChat dataset since LightGCN reports out-of-CUDA memory errors on an NVIDIA 2080Ti GPU (11G). In terms of GNP, we restrict the number of subgraph layers to 3. For computational efficiency, we sample the neighbors to accelerate the training and inference process. We set the sampling size of trajectory size as 25. We use Adam optimizer with a learning rate of 0.001,

Table 2. Comparison of hybrid recommendation performance. The percentages give the relative improvement over the best baseline (underlined). OOM denotes out-of-memory during the model training. The superscript * denotes statistical significance against the best baseline with $p < 0.05$.

Embedding	Model	CiteULike			XING			WeChat		
		Recall	Precision	NDCG	Recall	Precision	NDCG	Recall	Precision	NDCG
MF	DropoutNet	0.0780	0.0359	0.0654	0.1988	0.0712	0.1686	0.0519	0.0108	0.0315
	Heater	<u>0.0968</u>	<u>0.0479</u>	<u>0.0849</u>	<u>0.2190</u>	<u>0.0792</u>	<u>0.1921</u>	<u>0.0640</u>	<u>0.0130</u>	<u>0.0403</u>
	GNP	0.1404	0.0668	0.1187	0.2346	0.0866	0.2055	0.0659	0.0133	0.0443
	<i>Improv.</i>	45.04%*	39.46%*	39.81%*	7.12%*	9.34%*	6.98%*	2.97%*	2.31%*	9.93%*
M2V	DropoutNet	<u>0.1019</u>	<u>0.0443</u>	<u>0.0798</u>	<u>0.2159</u>	<u>0.0782</u>	<u>0.1764</u>	0.0685	0.0126	0.0401
	Heater	0.0779	0.0406	0.0687	0.1946	0.0716	0.1658	<u>0.0770</u>	<u>0.0139</u>	<u>0.0481</u>
	GNP	0.1292	0.0565	0.1172	0.2541	0.0937	0.2204	0.0804	0.0154	0.0535
	<i>Improv.</i>	26.79%*	27.54%*	46.87%*	17.69%*	19.82%*	24.94%*	4.42%*	10.79%*	11.23%*
LightGCN	DropoutNet	0.0933	0.0429	0.0795	0.2072	0.0756	0.1796	OOM	OOM	OOM
	Heater	<u>0.1136</u>	<u>0.0534</u>	<u>0.1018</u>	<u>0.2411</u>	<u>0.0873</u>	<u>0.2057</u>	OOM	OOM	OOM
	GNP	0.1631	0.0763	0.1411	0.2621	0.0954	0.2296	OOM	OOM	OOM
	<i>Improv.</i>	43.57%*	42.88%*	38.61%*	8.71%*	9.28%*	11.62%*	—	—	—

where the batch size is 1024. The coefficient of ℓ_2 normalization is set as $1e^{-5}$, and the randomized dropout ratio τ is set as 0.5. Tanh is used as the activation function. The size of the hidden layers and the output layer is set as 200, while the depth of MLP is set as 2 by default. Moreover, an early stopping strategy is performed by observing the AUC scores.

4.2 Main Results (RQ1)

Table 2 reports the performance comparison of GNP and baselines on the hybrid recommendation tasks, which evaluates the ability of cold-start models to rank top-k warm and cold-start items out of all items. We conduct our experiments on three different warm embeddings. From this table, we observe that GNP statistically outperforms all other baselines on all embeddings and all datasets with $p < 0.05$, demonstrating the consistent superiority of our GNP on hybrid recommendations. This further verifies that separating warm recommendation and cold-start recommendation models can exploit more potentialities from warm/cold-start recommendation models at the same time, resulting in better hybrid recommendation performance.

Further, to verify the pure-warm recommendation performance, in Table 3, we present how powerful are GNP and other baselines in recommending the top-k warm items out of all warm items. From this table, we highlight that GNP outperforms DropoutNet and Heater with statistically significant margins on all embeddings and all datasets. This further presents the necessity of solving cold-start problems with powerful GNN models.

Table 3. Comparison of warm recommendation tasks with the NDCG@K metric.

Embeddings	Models	CiteULike	XING	WeChat
MF	DropoutNet	0.0737	0.2167	0.0394
	Heater	<u>0.0934</u>	<u>0.2653</u>	<u>0.0489</u>
	GNP	0.1428	0.2757	0.0533
	<i>Improv.</i>	52.89%*	3.92%*	9.00%*
M2V	DropoutNet	0.0765	0.2211	0.0567
	Heater	<u>0.0857</u>	<u>0.2393</u>	<u>0.0629</u>
	GNP	0.1652	0.3157	0.0696
	<i>Improv.</i>	92.77%*	31.93%*	10.65%*
LightGCN	DropoutNet	0.0796	0.2506	OOM
	Heater	<u>0.1232</u>	<u>0.2930</u>	OOM
	GNP	0.1800	0.3398	OOM
	<i>Improv.</i>	46.10%*	15.97%*	–

Table 4. Comparison of average inferring time.

Model	CiteULike	XING	WeChat
DropoutNet	56 s	79 s	1259 s
Heater	73 s	97 s	1483 s
GNP (ours)	43 s	64 s	1054 s

4.3 Efficiency Study (RQ2)

In terms of the evaluation of model efficiency, we present the average inference times for DropoutNet, Heater, and GNP across the three datasets in question. As illustrated in Table 4, GNP demonstrates superior efficiency compared to both DropoutNet and Heater. This advantage can be attributed to the streamlined nature of GWarmer during the inference phase, where it operates with the simplicity of an inner product calculation. Moreover, the Patch Network is a lightweight architecture, which is efficient during representation patching. In contrast, DropoutNet and Heater necessitate the generation of embeddings through a more complex multi-layer perceptron (MLP) function, which inherently requires additional computational resources and time.

4.4 Ablation Study (RQ3)

In Figure 2, we present a comparative analysis of the hybrid recommendation performance between GNP, DropoutNet, and Heater, while varying the dropout ratio τ during training. Observing from this figure, GNP consistently outperforms its counterparts, DropoutNet and Heater, across the majority of situations. This observation further reinforces the consistent superiority of GNP in handling

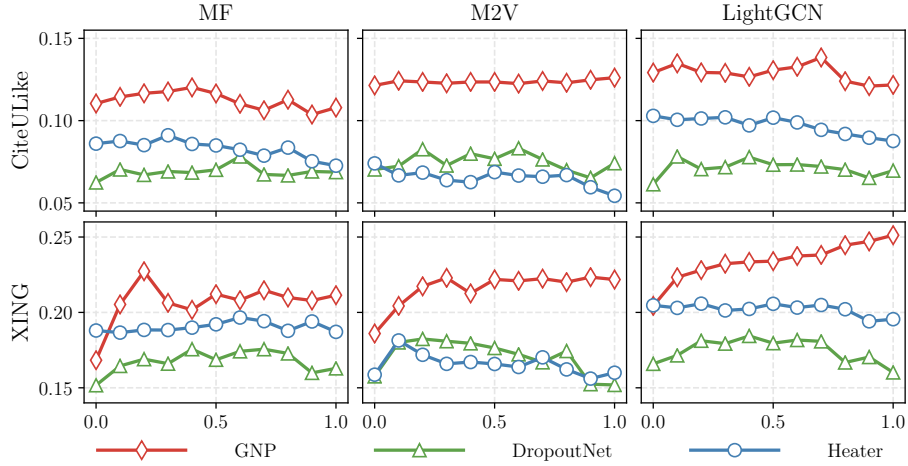


Fig. 2. Comparison of different training dropout ratio τ .

hybrid recommendation tasks. Especially, due to the Patching Network within GNP is specifically designed to address cold-start recommendations. Therefore, it is plausible that GNP achieves its optimal performance when the dropout ratio τ is set to 1. This configuration ensures that the model can fully leverage the Patching Network’s capabilities for cold-start items while maintaining its effectiveness for warm items.

5 Conclusion & Future Works

In this paper, we propose a simple but effective cold-start framework named Graph Neural Patching (GNP), which enhances the cold-start ability for GNN models with efficient modules (GWarmer and Patching Network). Specifically, GWarmer is an efficient GNN-based collaborative filtering module for the representation learning of warm users and items. Then, the crafted Patching Network is flexibly adopted to bolster the cold-start capabilities of GWarmer. From the results of extensive experiments, GNP is generally powerful on all datasets and embeddings in both the hybrid recommendation and the warm recommendation.

Moreover, the Patching Network that we propose serves as a versatile and efficient framework, enabling researchers to seamlessly integrate robust warm recommendation models with other effective cold-start models. This opens up a plethora of opportunities for future research endeavors, which could include: (i) designing more powerful patching cold-start models; (ii) flexibly combining with state-of-the-art warm recommendation models; (iii) investigating specialized patching models, such as large language models.

References

1. Bei, Y., Zhou, S., Tan, Q., Xu, H., Chen, H., Li, Z., Bu, J.: Reinforcement neighborhood selection for unsupervised graph anomaly detection. In: 2023 IEEE International Conference on Data Mining (ICDM). pp. 11–20. IEEE (2023)
2. Chen, H., Bei, Y., Shen, Q., Xu, Y., Zhou, S., Huang, W., Huang, F., Wang, S., Huang, X.: Macro graph neural networks for online billion-scale recommender systems. In: Proceedings of the ACM on Web Conference 2024. pp. 3598–3608 (2024)
3. Chen, H., Huang, W., Xu, Y., Sun, F., Li, Z.: Graph unfolding networks. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 1981–1984 (2020)
4. Chen, H., Wang, Z., Huang, F., Huang, X., Xu, Y., Lin, Y., He, P., Li, Z.: Generative adversarial framework for cold-start item recommendation. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2565–2571 (2022)
5. Chen, W., Bei, Y., Shen, Q., Chen, H., Huang, X., Huang, F.: Feedback reciprocal graph collaborative filtering. In: Proceedings of the 33rd ACM on Conference on Information and Knowledge Management (2024)
6. Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: Scalable representation learning for heterogeneous networks. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 135–144 (2017)
7. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864 (2016)
8. Guo, H., Tang, R., Ye, Y., Li, Z., He, X.: Deepfm: a factorization-machine based neural network for ctr prediction. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. pp. 1725–1731 (2017)
9. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in neural information processing systems (2017)
10. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M.: Lightgcn: Simplifying and powering graph convolution network for recommendation. Proceedings of the 43rd international ACM SIGIR conference on Research and development in Information Retrieval (2020)
11. Huang, F., Wang, Z., Huang, X., Qian, Y., Li, Z., Chen, H.: Aligning distillation for cold-start item recommendation. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1147–1157 (2023)
12. Huang, F., Yang, Z., Jiang, J., Bei, Y., Zhang, Y., Chen, H.: Large language model interaction simulator for cold-start item recommendation. arXiv preprint arXiv:2402.09176 (2024)
13. Huang, X., Li, J., Hu, X.: Label informed attributed network embedding. In: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining. pp. 731–739 (2017)
14. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with gpus. IEEE Transactions on Big Data **7**(3), 535–547 (2019)
15. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
16. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)

17. Lee, H., Im, J., Jang, S., Cho, H., Chung, S.: Melu: Meta-learned user preference estimator for cold-start recommendation. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1073–1082 (2019)
18. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
19. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 701–710 (2014)
20. Rao, N., Yu, H.F., Ravikumar, P.K., Dhillon, I.S.: Collaborative filtering with graph information: Consistency and scalable methods. *Advances in neural information processing systems* **28** (2015)
21. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: Bpr: Bayesian personalized ranking from implicit feedback. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence. pp. 452–461 (2009)
22. Rendle, S., Krichene, W., Zhang, L., Anderson, J.: Neural collaborative filtering vs. matrix factorization revisited. In: Proceedings of the 14th ACM Conference on Recommender Systems. pp. 240–248 (2020)
23. Sharma, K., Lee, Y.C., Nambi, S., Salian, A., Shah, S., Kim, S.W., Kumar, S.: A survey of graph neural networks for social recommender systems. *ACM Computing Surveys* **56**(10), 1–34 (2024)
24. Volkovs, M., Yu, G., Poutanen, T.: Dropoutnet: Addressing cold start in recommender systems. *Advances in neural information processing systems* **30** (2017)
25. Wang, X., He, X., Wang, M., Feng, F., Chua, T.S.: Neural graph collaborative filtering. In: Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval. pp. 165–174 (2019)
26. Wei, Y., Wang, X., Li, Q., Nie, L., Li, Y., Li, X., Chua, T.S.: Contrastive learning for cold-start recommendation. In: Proceedings of the 29th ACM International Conference on Multimedia. pp. 5382–5390 (2021)
27. Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K.: Simplifying graph convolutional networks. In: International conference on machine learning. pp. 6861–6871. PMLR (2019)
28. Wu, L., He, X., Wang, X., Zhang, K., Wang, M.: A survey on accuracy-oriented neural recommendation: From collaborative filtering to information-rich recommendation. *IEEE Transactions on Knowledge and Data Engineering* **35**(5), 4425–4445 (2022)
29. Wu, S., Sun, F., Zhang, W., Xie, X., Cui, B.: Graph neural networks in recommender systems: a survey. *ACM Computing Surveys* **55**(5), 1–37 (2022)
30. Zhang, Y., Bei, Y., Chen, H., Shen, Q., Yuan, Z., Gong, H., Wang, S., Huang, F., Huang, X.: Multi-behavior collaborative filtering with partial order graph convolutional networks. In: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 6257–6268 (2024)
31. Zhao, J., Zhou, Z., Guan, Z., Zhao, W., Ning, W., Qiu, G., He, X.: Intentgc: a scalable graph convolution framework fusing heterogeneous information for recommendation. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2347–2357 (2019)
32. Zhu, Z., Sefati, S., Saadatpanah, P., Caverlee, J.: Recommendation for new users and new items via randomized training and mixture-of-experts transformation. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1121–1130 (2020)