Adversarial Heterogeneous Graph Neural Network for Robust Recommendation

Lei Sang, Min Xu, *Member, IEEE*, Shengsheng Qian, *Member, IEEE*, and Xindong Wu, *Fellow, IEEE*

***Abstract*— Recommendation systems play a vital role in identifying the hidden interactions between users and items in online social networks. Recently, graph neural networks (GNNs) have exhibited significant performance gains by modeling the information propagation process in graph-structured data for a recommendation. However, existing GNN-based methods do not have broad applicability to heterogeneous graphs that integrate auxiliary data with diverse types. Moreover, graph structures are susceptible to noise and even unnoticed malicious perturbations, as perturbations from connected nodes can create cumulative effects on a target node in the graph. To enhance the robustness and generalization of GNN-based recommendations, we propose a new optimization model named Adversarial Heterogeneous Graph Neural Network for RECommendation (AHGNNRec). First, AHGNNRec learns user and item embeddings by exploring the distinct contributions of various types of interactions between users and items using a hierarchical heterogeneous graph neural network (HGNN). Second, to produce more robust embeddings for recommendations, we employ the adversarial training (AT) method to optimize the HGNN layers. AT is a min-max optimization training process where the generated adversarial fake nodes from normal nodes with intentional perturbations try to maximally deteriorate the recommendation performance. Following this, we learn about these adversarial user or item nodes by minimizing the impact of an additional regularization term for the recommendation. The experimental outcomes on two real-world benchmark datasets demonstrate the effectiveness of AHGNNRec.**

***Index Terms*— Adversarial training (AT), graph neural network (GNN), heterogeneous graph, recommendation.**

NOMENCLATURE

Notations	Description
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	Heterogeneous information graph (HIG).
$\mathcal{V}$	HIG node set.
$\mathcal{E}$	HIG link set.
$\mathcal{A}$	HIG node type set.
$\mathcal{R}$	HIG link type set.
ρ	Meta-path.
$\mathbf{e}_u$	User embedding.
$\mathbf{e}_i$	Item embedding.
$N^{(A_{t+1})}(i)$	Direct neighbor node set the type of A_{t+1}.
$\mathcal{N}_\rho(i)$	Sampled neighbor set obtained by meta-path ρ.
α_ρ	Attention weight for meta-path ρ.
$\mathbf{h_u}$	Learned user embedding.
$\mathbf{h_i}$	Learned item embedding.
$\mathbf{\Delta}$	Perturbations.

Manuscript received 13 October 2022; revised 27 February 2023 and 9 April 2023; accepted 17 April 2023. Date of publication 16 May 2023; date of current version 2 October 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62206002, in part by the Anhui Provincial Natural Science Foundation under Grant 2208085QF195 and Grant 2208085QF199, and in part by the Australia Research Council (ARC) Linkage Projects under Grant LP210100129. *(Corresponding author: Min Xu.)*

Lei Sang is with the School of Computer Science and Technology, Anhui University, Hefei, Anhui 230601, China (e-mail: sanglei@ahu.edu.cn).

Min Xu is with the Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW 2007, Australia (e-mail: Min.Xu@uts.edu.au).

Shengsheng Qian is with the Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: shengsheng.qian@nlpr.ia.ac.cn).

Xindong Wu is with the Key Laboratory of Knowledge Engineering with Big Data (the Ministry of Education of China), Hefei University of Technology, Hefei, Anhui 230601, China (e-mail: xwu@hfut.edu.cn).

Digital Object Identifier 10.1109/TCSS.2023.3268683

I. INTRODUCTION

RECENTLY, recommendation has become a crucial tool to assist users discover personalized information due to the unprecedented growth of data on a range of different platforms. Content-based [1], [2] and collaborative filtering-based [3], [4], [5], as two major recommendation methods, have been widely studied in both academia and industry. However, both these methods have certain drawbacks: 1) content-based methods merely measure item relevance using specific data analysis, meaning that they cannot directly reflect users' general interests and 2) collaborative filtering-based methods are faced with the problem of data sparsity and cold-start [6].

By organizing user–item interactions in a graph structure, significant improvements have been made in recent years when it comes to learning embeddings of graph structural data for downstream recommendation tasks [7], [8], [9]. The most prominent technique involves GNN-based models, which can mine higher-order interactions among users and items and has achieved relatively good performance in recommendation [10], [11]. For example, PinSage [12] applies graph convolutional networks to the pin-board bipartite graph in Pinterest, updating the embedding of each pin using both graph structure and node feature information. Multi-modal graph convolution network (MMGCN) [13] learns modality-specific embeddings of users and microvideos with GNN to capture user preferences. The key idea behind these graph-based recommendation methods is to apply the iterative propagation operation to each node in the graph; here, propagation operation involves learning target node embeddings by aggregating local neighbors' context features in the previous layer. By stack multilayer propagation operation, we can exploit high-order interactions among users and items, thereby generating high-quality item or user embeddings for recommendation.

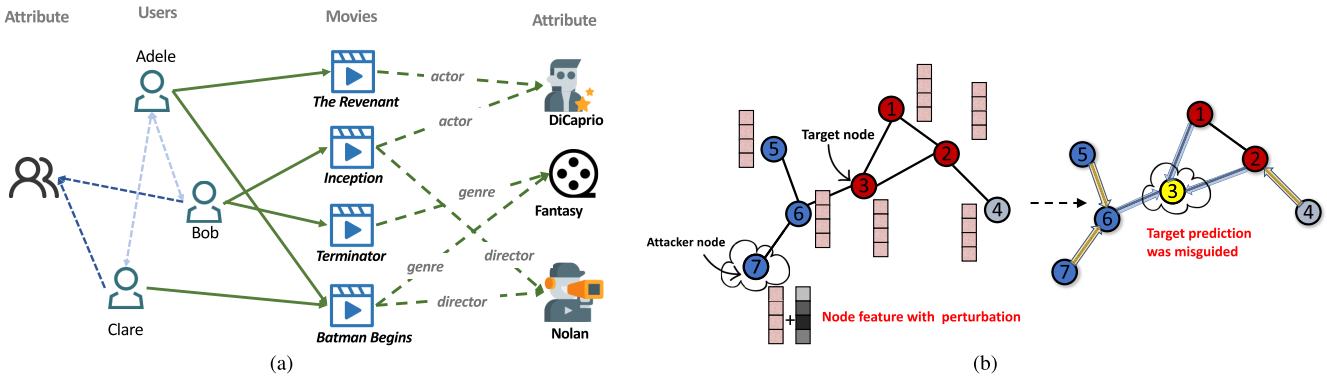


Fig. 1. (a) We try to encode high-order interdependency of context into the comprehensive user and item embedding for recommendation purposes. (b) Small perturbations of the node feature lead to misclassification of the target.

Despite the promising performance, however, these GNN-based recommendation methods cannot learn comprehensive embeddings for recommendation due to the following two reasons. First, most current neural graph-based recommendation methods lack the ability to encode the crucial heterogeneous auxiliary data explicitly and are commonly applied to homogeneous graphs with only one type of node. Heterogeneous information graphs (HIGs) are important tools to alleviate the cold start and sparsity problem in recommender systems, which have great flexibility to mine various patterns of interactions and can therefore reveal the hidden interdependencies among users and items [14], [15], [16], [17]. For example, the relations between two movies can be inferred from: 1) "Movie–User–User–Movie" (MUUM) and 2) "Movie–Category–Movie", as shown in Fig. 1(a). These meta-paths capture the semantic relations of: 1) being watched by friends and 2) belonging to the same genre. Different from homogeneous graphs, most nodes in HIG did not have a direct connection to all types of neighbors. Moreover, different types of nodes may have diverse impacts on the node embeddings in HIG [18], [19]. However, recent studies of neural graph-based recommendation have focused primarily on homogeneous graphs and rarely consider the impacts of various node types. Second, GNN-based recommendation methods are vulnerable to unnoticeable deliberate perturbation attacks on graph node features, similar to other neural networks [20], [21], [22], [23]. For example, malicious users can create fake profiles by connecting to targeted users or connecting to fake users, to mislead neural graph-based recommendation models. We accordingly argue that the effects of perturbation attacks could make more serious harm to the training of GNNs than standard neural networks due to the propagation process of GNNs, as the propagation of information might lead to cascading effects, where manipulating a single node will affect many others [24], [25]. For example, in Fig. 1(b), we deliberately apply perturbations to the features of node 7. As a result, the neural graph-based model is fooled into making incorrect predictions regarding the more distant node 3 due to the propagation effect. In a real recommendation scenario, small noise or perturbations may occur frequently (such as the updating of node features) but should not have a noticeable impact on predictions. Hence, we argue that it is necessary to stabilize the training process of neural graph-based recommendation models.

The robust recommendation aims to ensure that recommendation models are resilient to various imperfect environments. As recommendation systems can influence user purchasing and browsing behavior, they bring economic benefits. Therefore, many malicious users attempt to control recommendation systems through anomalous attacks to increase item sales or traffic or even to disrupt the system and render it incapable of generating effective recommendations. The GraphRfi method [26] generates stable recommendations by using an auxiliary fraud detection task while training graph recommendation models. For shilling attacks encountered in recommendation systems, many methods extract useful features from the original data (such as the average rating deviation, neighbor similarity, etc.) and then train a supervised detection model to identify the shilling attack data and remove it from the dataset [27], [28]. However, such methods cannot integrate rich auxiliary information of both users and items well for attacker detection, which may lead to false negatives and allow some attacking users to evade detection. Besides, existing work on robust recommendation has rarely targeted complex, multisource, heterogeneous graph-structured data, and has not considered the cascading amplification effect of iterative propagation of anomalous attacks in the graph structure.

To enhance the robustness and generalization of GNN-based recommendations, we propose a new optimization model named Adversarial Heterogeneous Graph Neural Network for RECommendation (AHGNNRec). First, we devise a new GNN for heterogeneous graphs to obtain the embedding of users and items. specifically, a meta-path-based random walk strategy is designed to sample fixed-size neighbors of each node in the HIG and group them according to node types. A hierarchical aggregation method is then utilized to measure the different contributions of semantic type and generate user and item node embedding for recommendations. Next, to further improve the robustness of the learned embedding from the HIG, an adversarial training (AT) [21], [29], [30] method is applied to the above neural graph-based recommendation. AT can be seen as a dynamic regularization technique that deliberately applies the carefully designed perturbations to the training process of neural networks. AT be featured with a min-max optimization

process: the perturbations are learned to maximize the final prediction loss with the user or item embedding learned from GNNs, followed by learning over these adversarial user or item nodes by minimizing the impact of the additional regularization term for recommendations. By adopting this approach, AHGNNRec can defend the worst-case perturbations during node embedding learning and improve the recommendation robustness, such that the perturbations have a relatively small influence on the model's prediction.

Our contributions can be summarized as follows.

1) We propose a hierarchical aggregation-based heterogeneous graph neural network (HGNN) to explore the different contributions of multiple type user–item interactions.
2) We apply adversarial learning into the training phase of the neural graph recommendation model, which can improve the robustness of recommendation prediction by dynamically constructing adversarial examples.
3) Extensive experiments have been conducted on two real-world datasets, the results of which demonstrate the efficacy of AHGNNRec over strong baselines.

II. Preliminaries

With the booming of online social networks, a huge amount of information has been produced for recommendation, including various types of objects (e.g., User (U), Movie (M), Genre (G), and Actor (A)) and their semantic relationships (e.g., "friends" relations among users, "watching" relations between users and movies, and "attribute" relations between movies and directors/actors) in movie recommendation. This auxiliary information is critical to alleviating the sparsity and cold-start problem associated with simple user–movie interactions. A heterogeneous information graph (HIG) is a powerful tool in organizing this auxiliary information. Users and movies with few or no interactions before can establish new connections with the help of various types of paths in an HIG. HIG can be defined as follows.

Definition 1 (HIG): An HIG is denoted as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, which consists of a node-type mapping function $\phi : \mathcal{V} \to \mathcal{A}$ and a link-type mapping function $\varphi : \mathcal{E} \to \mathcal{R}$. $\mathcal{A}$ and $\mathcal{R}$ are the sets of predefined nodes and link types, respectively, and satisfy $|\mathcal{A}| + |\mathcal{R}| > 2$.

As shown in Fig. 2, the **network schema** is used to demonstrate the high-level topology of an HIG, which describes the node types and their interaction relations. In HIG, nodes can be connected via multiple types of semantic paths, which are defined as **meta-paths**.

Definition 2 (Meta-Path): Given an HIG, the meta-path ρ is defined in the form of $\mathcal{A}_1 \xrightarrow{\mathcal{R}_1} \mathcal{A}_2 \xrightarrow{\mathcal{R}_2} \cdots \xrightarrow{\mathcal{R}_l} \mathcal{A}_{l+1}$ (abbreviated as $\mathcal{A}_1\mathcal{A}_2, \ldots, \mathcal{A}_{l+1}$), which denotes a composite relation $\mathcal{R}_1 \circ \mathcal{R}_2 \circ, \ldots, \mathcal{R}_l$ between nodes $\mathcal{A}_1$ and $\mathcal{A}_{l+1}$, where $\circ$ is a composition operator applied to relations.

Example 1: There exist multiple specific paths under the meta-path, which is called a path instance denoted by ρ. We can connect two movies with different meta-paths, for example, "Movie–Actor–Movie" (MAM) and MUUM. Commonly, different meta-paths can reveal the different interaction semantic information of two movies. Path "MAM" demonstrates that the same actor acts in two movies, while path "MUUM" demonstrates two movies watched by two users with friend relationship.

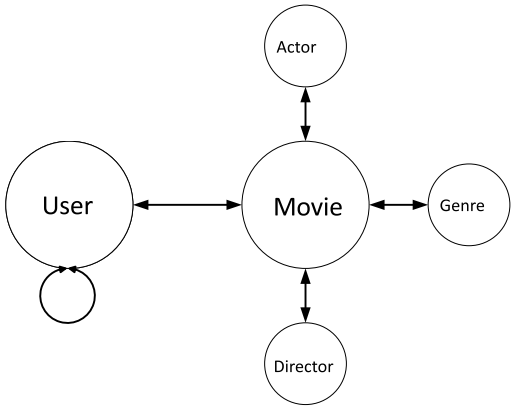


Fig. 2. Network schemas of a heterogeneous graph used for our social movie recommendation scenario.

III. Methodology

Fig. 3 shows the overall framework of our AHGNNRec model. First, AHGNNRec adopts a meta-path-based random walk to sample the type-specific neighbors and then a hierarchical aggregation mechanism is applied to the GNNs in order to learn comprehensive user and item embeddings. Moreover, we apply AT by dynamically constructing adversarial perturbations to further enhance the generalization and robustness of neural graph-based recommendation. Note that the perturbations $\mathbf{\Delta}$ are enforced on the embedding of attacked fake nodes. The propagation and prediction models are trained jointly for final recommendation prediction.

A. Embedding Layer

Given the input graph data, the embedding layer obtains the low-dimensional hidden representations of nodes, which can characterize the intrinsic properties of users and items [31], [32]. We denote a user u (an item i) with an embedding vector $\mathbf{e}_u \in \mathbb{R}^d$ ($\mathbf{e}_i \in \mathbb{R}^d$), where d is the size of embedding. This can be conceptualized as transformation matrices that map the user or item, respectively, into the hidden space, as follows:

$$\mathbf{E} = [\underbrace{\mathbf{e}_{u_1}, \ldots, \mathbf{e}_{u_N}}_{\text{users embeddings}}, \underbrace{\mathbf{e}_{i_1}, \ldots, \mathbf{e}_{i_M}}_{\text{item embeddings}}]. \tag{1}$$

B. Meta-Path-Based Heterogeneous Neighbors Sampling

Most neural graph network studies mainly focus primarily on homogeneous networks, such as GraphSAGE [33] or GAT [34]. However, these methods cannot be directly deployed to HIG due to the various types of neighbors around the target node. Moreover, the neighbor size grows exponentially as the L-layers increase, which might introduce noise and consequently decrease accuracy. And these methods can also be biased by various neighbor sizes, where some core nodes and hub nodes are usually connected to many other nodes.

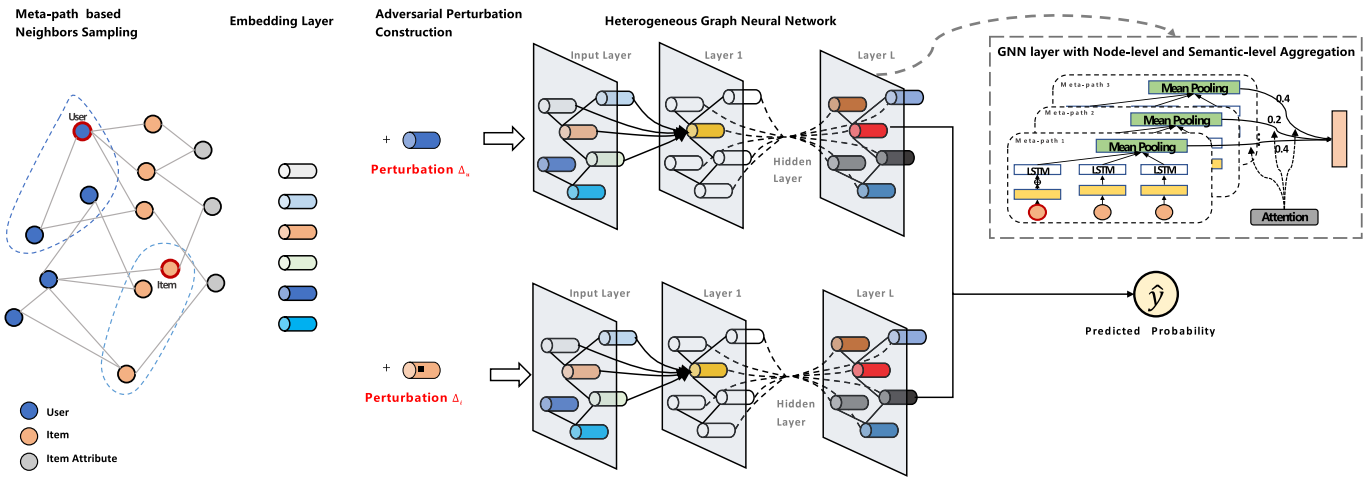


Fig. 3. Overall framework of AHGNNRec. We first sample neighbors for users and items with meta-path based random walks. The sampled neighbors fed into the graph-based propagation layer. Note that the perturbations $\mathbf{\Delta}$ are enforced on the embedding of attacked fake nodes. The propagation and prediction model are trained jointly for final recommendation prediction.

Here, a Meta-Path-based Random Walks (MPRW) method is designed over the heterogeneous graph to sample heterogeneous neighbors. Given an HIG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a meta-path $\rho : A_1 \xrightarrow{R_1} A_2 \xrightarrow{R_2} \cdots A_t \xrightarrow{R_t} A_{t+1} \cdots \xrightarrow{R_l} A_{l+1}$, the transition probability of the next walk is as follows:

$$P(n_{t+1}|n_t, \rho) = \begin{cases} \frac{1}{|N^{(A_{t+1})}(v)|}, & \substack{(n_t, n_{t+1}) \in \mathcal{E} \\ \phi(n_{t+1}) = A_{t+1}} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where $n_t \in A_t$ is the tth node in the random walk sequence, while $N^{(A_{t+1})}(n_t)$ denotes the direct neighbor node set for n_t with the type of A_{t+1}. A meta-path-based random walk will iteratively follow the predefined path pattern iteratively until it reaches the predefined fixed number. Specifically, we denote the sampled neighbor set obtained by meta-path ρ as $\mathcal{N}_\rho(i)$ for node i. (Note that the target node's neighbor set includes itself.) This strategy can avoid the aforementioned issues for the following reasons: 1) MPRW can ensure that the different semantic interactions among various types of nodes can be well retained and 2) neighbor nodes are divided into groups according to the type such that type-based aggregation process can be used.

C. Hierarchical HGNN

Next, we devise a new graph neural network model for HIG to learn the representation of users and items. An HGNN takes a heterogeneous graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ as input, which is a graph that contains multiple types of nodes and edges. And the user or item node features are defined as $\mathbf{E}$ in the embedding layer. The output of an HGNN is the user and item node embedding that captures the underlying structure and features of the heterogeneous graph, which can then be used for downstream recommendation tasks. The output of HGNN is obtained through the aggregation process, which involves combining information from different types of nodes and edges in the graph. During the aggregation process, each node aggregates information from its neighbors according to a specific aggregation function. The aggregation function typically takes into account the types of nodes and edges involved in the message-passing process, as well as the features associated with these nodes and edges.

The key aggregation process of HGNN has a hierarchical structure: node-level and semantic-level. First, we select all the neighbors obtained by the same kind of meta-path random walk and then combine them to learn the type-specific node representation with node-level aggregation. Subsequently, we use semantic-level attention to obtain the optimally weighted aggregation of the semantic-specific node representation for recommendations.

1) Node-Level Aggregation: For each node in the graph, we use the MPRW-based sampling strategy to generate neighbor sets through various meta-path types. And the type-specific node representation is obtained as follows:

$$\boldsymbol{z}_\rho^i = \sigma(g(\{\boldsymbol{e}_j, \forall j \in \mathcal{N}_\rho(i)\}) \tag{3}$$

where $g(\cdot)$ is the aggregation function. Many aggregation functions could be used (such as max pooling mean average and fully connected neural network). Here, we employ Bi-LSTM to aggregate the neighbor node with the same type, and then all hidden states of Bi-LSTM are averaged as the general aggregated representation. Thus, the representation of node $\mathbf{z}_\rho^i$ for meta-path ρ can be calculated as follows:

$$\mathbf{z}_\rho^i = g(\text{LSTM}\{\boldsymbol{e}_j, \forall j \in \mathcal{N}_\rho(i)\}). \tag{4}$$

2) Semantic-Level Attention Aggregation: In practice, a very important issue regarding the aggregation process is that different neighbors may be sampled with different meta-paths, which reflect the various types of hidden semantic information among users and items. Meta-paths can reveal various interaction semantics, and a more comprehensive node embedding can be obtained by fusing these semantics. To address the problem of meta-path selection and semantic fusion in HIG, we propose a semantic-level attention to adaptively learn the different contributions of meta-paths and then fuse them for final representation. Taking P types of semantic-specific node embeddings learned from the node-level attention as input, the

learned weights of each meta-path $(\alpha_{\rho_0}, \alpha_{\rho_1}, \ldots, \alpha_{\rho_P})$ are used to indicate the contribution of the ρ-type neighbors to target node, which is defined as follows:

$$(\alpha_{\rho_0}, \alpha_{\rho_1}, \ldots, \alpha_{\rho_P}) = \text{att}_{\text{sem}}(\mathbf{Z}_{\rho_0}, \mathbf{Z}_{\rho_1}, \ldots, \mathbf{Z}_{\rho_P}). \tag{5}$$

Here, att_{sem} denotes the semantic-level attention neural network model. Thus, the output embedding for node i is formulated as follows:

$$\mathbf{h_i} = \mathbf{e_i} + \sum_{\mathbf{t=1}}^{\mathbf{P}} \alpha_{\rho_\mathbf{t}} \cdot \mathbf{Z}_{\rho_\mathbf{t}} \tag{6}$$

where $\mathbf{h_i}$ denote the final output representation of node i, while $\mathbf{Z}_{\rho_t}$ is the neighbor-aggregated embedding for meta-path ρ_t. We sum the weighted semantic-specific node representation, which can be seen as the contribution of each meta-path for the final representation.

Moreover, the attention matrix α_{ρ_t} indicating the importance of different embeddings is defined as follows:

$$\alpha_{\rho_t} = \frac{\exp(\text{ReLU}(a^T[\boldsymbol{e}_i || \boldsymbol{Z}_t]))}{\sum_t^P \exp(\text{ReLU}(a^T[\boldsymbol{e}_i || \boldsymbol{Z}_t]))} \tag{7}$$

where $||$ denotes the concatenation operation and $a \in \mathbb{R}^{2\text{D}}$ is a weight vector, which can be explained as a single-layer feedforward neural network. All ReLUs in the attention are leaky, with a slope of 0.2. To get the weight of each meta-path type, we first process semantic-specific embedding with means of a nonlinear transformation (e.g., one-layer MLP: $a \in \mathbb{R}^{2\text{D}}$). We then obtain the importance of the semantic-specific embedding by measuring the similarity of the transformed embedding with a target node embedding $\mathbf{e}_i$. As is evident, the higher the value of α_{ρ_t}, the more importance of meta-path ρ_t.

D. Prediction Probability With BRP Loss

We construct the user–item prediction model between the user and the item using the matrix factorization (MF) method. MF characterizes both users and items with real-valued low-dimensional hidden embedding, which generates the recommendation results by measuring the prediction probability with the inner product

$$\hat{y}_{ui} = \mathbf{h}_u \mathbf{h}_i. \tag{8}$$

To train this model, we apply Bayesian personalized ranking (BPR) loss to get final prediction. The core idea of BPR is to use a ranking-based approach to optimize the model parameters. Specifically, BPR tries to maximize the probability that a user will prefer an item that they have interacted with over a randomly sampled item that they have not interacted with. The loss function is defined as the negative log-likelihood of observing the preferences of users for items. BPR is a pairwise personalized ranking loss that maximizes the margin between an observed positive interaction and its unobserved negative interactions. Formally, the loss function of BPR is minimized as follows:

$$L(\mathcal{D}|\Theta) = \sum_{(u,i,j)\in\mathcal{D}} -\ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}) + \lambda_\Theta ||\Theta||^2 \tag{9}$$

where $\mathcal{D}$ are the training instances set: $\mathcal{D} := \{(u, i, j)|i \in \mathcal{K}_u^+ \wedge j \notin \mathcal{K}_u^+\}$, and $\mathcal{K}_u^+$ are item sets that have interacted before with user u, while λ_Δ denote the regularization hyperparameters employed to balance the two terms. The loss function consists of two terms. The first term measures the likelihood of the observed preferences of users for items. The second term is the L2 norm regularization term that helps prevent overfitting by penalizing large parameter values. The BPR loss function is designed to optimize for pairwise ranking of items rather than pointwise ranking. This is because pairwise ranking provides a more efficient way to capture user preferences and reduce the effects of biases that may be present in the data. The BPR loss function can be optimized using stochastic gradient descent (SGD) or other gradient-based optimization algorithms. During training, the model learns the user and item embeddings that minimize the BPR loss function. These embeddings can then be used to make personalized recommendations for users.

E. AT for Robust Recommendation

Recent advances in AT have successfully improved the robustness of the deep neural network by feeding generated adversarial perturbations into the training process [35], [36], [37], [38]. Inspired by the effectiveness of AT, we propose to adopt an AT method to defend against fake input feature attacks on the graph-based recommender model. This method has two key points: 1) generating slight but carefully crafted adversarial examples that degrade the model performance by adding perturbations to the input node feature and 2) training the model to defend against the damage brought by these perturbations. We aim to design a new optimization objective for the recommendation model, which can not only generate fairly good prediction, but also defend the adversarial perturbation attacks.

1) Adversarial Perturbation Construction: In more detail, the goal of our adversary construction is to attack a specific target node, change it into a new fake node, and then change its final recommendation prediction result. The perturbations are usually applied to model input features; here, however, we apply perturbations to the input item embedding $\mathbf{e}_i$

$$\hat{y}'_{ui} = \mathbf{h}'_u \boldsymbol{h}'_i = f(\boldsymbol{e}_u + \boldsymbol{\Delta}_u) f(\boldsymbol{e}_i + \boldsymbol{\Delta}_i) \tag{10}$$

where $f()$ denotes the heterogeneous GNN layer, $\boldsymbol{\Delta}_u$ is the perturbations added on user u, and $\boldsymbol{\Delta}_i$ is the perturbations applied on item i.

We construct adversarial perturbation to find the best perturbations $\boldsymbol{\Delta}$ that can damage the performance of the recommendation model to the greatest extent, which is also known as the worst-case perturbations [24], [29]. Taking item i as an example, the formulation of the adversary perturbation construction is as follows:

$$\begin{aligned} \textbf{max: } \boldsymbol{\Delta}_i^* &= \arg\max_{\boldsymbol{\Delta}} L(\mathcal{D}|\Theta)' + \sum_{i=1}^{N} KL\big(\boldsymbol{h}'_i, \boldsymbol{h}_k, \forall k \in \mathcal{N}(i)\big) \\ &= \arg\max_{\boldsymbol{\Delta}} \sum_{(u,i,j)\in\mathcal{D}} -\ln \sigma\big(\hat{y}'_{ui} - \hat{y}'_{uj}\big) \\ &\quad + \lambda \sum_{i=1}^{N} \sum_{k\in\mathcal{N}(i)} KL\Big(f(\boldsymbol{e}_i + \boldsymbol{\Delta}_i), f(\boldsymbol{e}_k)\Big), \text{ where} \\ &\quad ||\Delta_i|| \le \epsilon \end{aligned} \tag{11}$$

Algorithm 1 SGD Learning Algorithm for AHGNNRec

Input: Training data $\mathcal{D}$, adversarial noise level ϵ, adversarial regularizer α, smooth regularizer β, learning rate η;
Output: Model parameters Θ;

```
Initialize Θ from HGNN+BPR(HGNNRec) ;
while Stopping criteria not met do
    Randomly draw (u, i, j) from D ;
    // Adversarial perturbations Construction
    Δ_adv ← Equation (15) ;
    // Updating model parameters
    Θ ← Equation (16) ;
end
return Θ
```

where $\mathbf{\Delta}_i$ is the small but carefully crafted graph perturbations, which can be added to the clean item example feature $\mathbf{e}_i$ to construct a graph adversarial example. Since our model is trained to minimize the BPR loss and therefore facilitates better recommendation prediction [see (9)], it is natural that the first term should be designed to go in an opposite optimization direction for the perturbations, namely to **maximize** the BPR loss with the help of perturbation $\mathbf{\Delta}$. The goal of the second term is to smooth the predictions with graph structure; this is due to the fact that closely connected nodes should have close embeddings in the graph, where KL() is the Kullback–Leibler divergence to measure the divergence between adversarial and clean embeddings of normal neighbor nodes [24].

2) Model Optimization: To train our recommendation model robust to the adversarial perturbations, we not only need to minimize the original BPR prediction loss, but also need to **minimize** the adversary's objective function. We define the optimization objective for the model as follows:

$$\begin{aligned}\textbf{min: } \Theta^* &= \arg\min_{\Theta} L(\mathcal{D}|\Theta) + L(\mathcal{D}|\Theta)' \\ &\quad + \sum_{i=1}^{N} \mathrm{KL}\big(\boldsymbol{h}_i', \boldsymbol{h}_k, \forall k \in \mathcal{N}(i)\big) \\ &= \arg\min_{\Theta} \sum_{(u,i,j)\in\mathcal{D}} -\ln\sigma(\hat{y}_{ui} - \hat{y}_{uj}) \\ &\quad - \alpha \ln\sigma\big(\hat{y}'_{ui} - \hat{y}'_{uj}\big) \\ &\quad + \beta \sum_{i=1}^{N} \sum_{k\in\mathcal{N}(i)} \mathrm{KL}\big(f\big(\boldsymbol{e}_i + \mathbf{\Delta}_i^*\big), f(\boldsymbol{e}_k)\big) \end{aligned} \tag{12}$$

where α and β are hyperparameters used to balance the weights of the adversarial example on the recommendation prediction. When α and β are set to 0, the model degrades to a pure GNN propagation model and the worst-case adversary example has no impact on model optimization. In this formulation, the loss of adversary optimization $L(\mathcal{D}|\Theta)'$ and $\sum_{i=1}^{N} KL\big(\boldsymbol{h}_i', \boldsymbol{h}_k, \forall k \in \mathcal{N}(i)\big)$ can be seen as regularization term [24], [39], [40] to train the robust to adversary example.

To unify the two processes, AT in our approach can be formulated as a **min-max** optimization process, to find the best solution to the worst-case optimum. More specifically, the min-player player is the optimization of model parameters Θ^*, and the max-player player is the construction of perturbations Δ_i^*

$$\begin{aligned}\Theta^*, \Delta_i^* &= \arg\min_{\Theta}\max_{\Delta} L(\mathcal{D}|\Theta) + L(\mathcal{D}|\Theta)' \\ &\quad + \sum_{i=1}^{N} KL\big(\boldsymbol{h}_i', \boldsymbol{h}_k, \forall k \in \mathcal{N}(i)\big), \text{ where} \\ &\quad ||\Delta_i|| \le \epsilon. \end{aligned} \tag{13}$$

The min-max optimization method requires the specification of three more hyperparameters: ϵ, α, and β, which are crucial to the model training and need to be carefully tuned. In more detail, if these values are too large, the model can defend the adversarial perturbations, albeit at the cost of damaging the recommendation performance; moreover, values that are too small will make the adversarial perturbation cannot play its role and limit the improvement of the model robustness. A reasonable interpretation of this formulation can be summarized as: the inner maximization optimization is finding the worst-case samples for the given model, and the outer minimization optimization is training a model robust to adversarial examples.

3) Learning Algorithm:

a) Adversary approximation: For each training instance (u, i, j), it is nontrivial to learn the optimal solution of the perturbation $\mathbf{\Delta}$. Inspired by the *linear approximation* method for standard AT proposed in [24] and [29], we also design a linear approximation method to calculate the graph adversarial perturbations in AHGNNRec. Linear approximation is a common technique used in AT to approximate the behavior of the model near the decision boundary. It is clear that to maximize a linear function, the closed-form solution is to move the variables toward the direction of their gradients. Taking the ϵ-constraint into account, we can obtain the solution for adversarial perturbations as follows:

$$L_{\text{adv}}((u,i,j)|\Delta_i) = L((u,i,j)|\hat{\mathbf{\Theta}})' + \sum_{k\in\mathcal{N}(i)} KL\big(\boldsymbol{h}_i^{L'}, \boldsymbol{h}_k^{L},\big) \tag{14}$$

$$\mathbf{\Delta}_{\text{adv}} \approx \epsilon \frac{\boldsymbol{g}}{\|\boldsymbol{g}\|}, \quad \text{where } \boldsymbol{g} = \frac{\partial L_{\text{adv}}}{\partial \Delta_i} \tag{15}$$

where $\boldsymbol{g}$ is the gradient. Note that the model's parameter $\hat{\mathbf{\Theta}}$ is a constant set. In linear approximation, the adversarial perturbation is assumed to be small enough so that the model's behavior can be approximated using a linear function. This linear function is typically the gradient of the loss function with respect to the input data. By adding a small multiple of the gradient to the input data, the adversarial example can be generated. The linear approximation technique is often used in iterative AT, where the model is trained using a combination of clean and adversarial examples. In each iteration, the adversarial examples are generated using the linear approximation technique, and the model is trained on both the clean and adversarial examples.

b) Learning model parameters: This step updates the model parameters by minimizing (12). Since the perturbations Δ are fixed in this step, the problem becomes a conventional minimization problem and can be approached as follows:

$$L_{\text{AHGNNRec}}((u,i,j)|\Theta) = L((u,i,j)|\Theta) + L((u,i,j)|\Theta)' + \sum_{k\in\mathcal{N}(i)} KL\left(\boldsymbol{h}_i^{L'}, \boldsymbol{h}_k^{L},\right). \quad (16)$$

We can then obtain the following SGD update rule for θ:

$$\theta = \theta - \eta \frac{\partial L_{\text{AHGNNRec}}}{\partial \theta} \quad (17)$$

where η denotes the learning rate, which can be parameter-dependent if adaptive SGD methods are used; we use the Adagrad in our experiments.

Algorithm 1 summarizes the overall solution of the entire adversarial framework. In each training step (lines 3–5), we first randomly draw an instance (u, i, j). The update rules for adversarial perturbations and model parameters are then processed alternately. Moreover, we initialize the model parameters Θ by pretraining GNN + BRP (line 1), rather than employing random initialization. One reason is that training with adversarial perturbations will alleviate the overfitting of the neural network. When the model is underfitting, a normal training process is sufficient to produce better parameters [39].

IV. Experiments

To evaluate the performance of our proposed method, we conducted a series of experiments to compare nine other baseline recommendation methods. We first describe experimental settings in Section IV-A and then conduct a detailed comparison with nine other baseline recommendation methods in Section IV-B. To justify the designs in AHGNNRec and reveal the reasons for its effectiveness, we perform ablation studies and analyses in Section IV-C. The hyperparameter study is finally presented in Section IV-D.

A. Experimental Settings

1) Datasets Description: The proposed AHGNNRec is evaluated on two widely used benchmark datasets. These two heterogeneous datasets represent two recommendation scenarios for videos and businesses, respectively.

1) *YouTube*[1]*:* For the video recommendation scenario, we use a cross-network YouTube dataset with user account linkage between YouTube and Twitter [41]. This dataset includes 143 259 Google+ users, 11 850 of whom provided both their Twitter and YouTube accounts. To reduce noise, we retain only the top 1000 most frequently used tags in our experiment.
2) *Yelp*[2]*:* Yelp dataset contains all kinds of reviews from the Yelp website. This dataset consists of 16 239 users and 14 282 local businesses. There are also 198 397 ratings from value 1 to 5. There are five types of nodes: user, business, compliment, city, and category. We extract subsets of entities from Yelp to build the HIG for recommendations.

[1]http://www.nlpr.ia.ac.cn/mmc/homepage/myan/dataset.html
[2]http://www.yelp.com/dataset_challenge

2) Compared Methods: We compare the proposed AHGNNRec with several strong baselines, which can fall into three categories: CF and Content-based (LFM, BPR-MF, AMF, and Bi-LSTM), HIG-based (SemRec and HERec), and GNN-based (NGCF, LightGCN, and HGNNRec) methods.

1) *LFM [42]:* A collaborative filtering approach combines neighborhood and SVD-based latent factor model, which integrate both explicit and implicit user interaction.
2) *BPR-MF [43]:* An MF-based BPR method, which tries to maximize the margin for the user's positive observed pairs and minimize the margin for nonobserved user–item pairs at the same time.
3) *AMF [40]:* Adversarial MF, which implements adversarial BPR loss on MF by adding adversarial perturbations to user and item embeddings.
4) *Bi-LSTM [44]:* A sequential recommendation model uses bidirection long short-term memory units as the building units.
5) *SemRec [45]:* An HIG-based recommendation model that explores the weighted meta-path semantic information to learn the different impact of paths on users' preferences.
6) *HERec [15]:* Another HIG-based recommendation model that learns the user and item node embedding for recommendation by combining different meta-path semantics.
7) *CDPRec [7]:* Context-dependent propagating-based recommendation uses a meta-path-based random walker to extract a homogeneous graph from the heterogeneous graph, which then applies GNNs to learn user and item embedding for recommendations.
8) *NGCF [13]:* Neural Graph Collaborative Filtering (NGCF) is a neural graph-based recommendation model, which mines high-order interactions in the graph by iteratively learning the propagation embedding.
9) *LightGCN [8]:* Different from NGCF, LightGCN discards the feature transformation operation and obtains the final embedding only by summing the weighted embeddings learned at all layers.
10) *HGNNRec:* A simplification version of AHGNNRec without AT. HGNNRec model stacks two HGNN layers to learn the node representation. A similar BPR loss is then adopted to obtain the final recommendation prediction.

3) Parameter Settings and Evaluation Protocol: Our AHGNNRec model is implemented with Pytorch. 80% of the users in the dataset are used for training purposes, while both the validation and test sets contain 10% of users. The pairwise learning framework trains the recommendation model with the help of ranking computation between target positive rating and negative ratings. To improve the computational efficiency, we randomly sample 50 users' noninteractive items as negative samples. We adopt two representative ranking performance measures—Normalized Discounted Cumulative Gain at rank K (NDCG@K) and Hit Ratio at rank K (HR@K)—as the evaluation metrics to evaluate the top-K recommendation performance. The formula for Normalized Discounted Cumulative

TABLE I
PERFORMANCE COMPARISON BETWEEN AHGNNREC AND THE BASELINES

	YouTube						Yelp					
	HR@K			NDCG@K			HR@K			NDCG@K		
	K=10	K=50	K=100	K=10	K=50	K=100	K=10	K=50	K=100	K=10	K=50	K=100
LFM	0.1425	0.1585	0.1874	0.0024	0.0038	0.0046	0.1415	0.3025	0.4238	0.0317	0.0534	0.0765
BPR-MF	0.1489	0.1613	0.1928	0.0163	0.0223	0.0283	0.1432	0.3186	0.4365	0.0336	0.0578	0.0782
AMF	0.1533	0.1627	0.2011	0.0203	0.0298	0.0324	0.1602	0.3512	0.4628	0.0361	0.0601	0.0814
Bi-LSTM	0.1543	0.1711	0.2345	0.0231	0.0314	0.0349	0.1579	0.3609	0.4937	0.0358	0.0621	0.0852
SemRec	0.1524	0.1719	0.2376	0.0321	0.0385	0.0467	0.1547	0.3725	0.5136	0.0386	0.0679	0.0871
HERec	0.1549	0.1742	0.2457	0.0347	0.0397	0.0492	0.1725	0.4117	0.5239	0.0402	0.0683	0.0893
CDPRec	0.1603	0.1814	0.2546	0.0352	0.0417	0.0513	0.1903	0.4196	0.5525	0.0439	0.0761	0.0937
NGCF	0.1587	0.1794	0.2513	0.0351	0.0411	0.0507	0.1888	0.4191	0.5490	0.0422	0.0714	0.0919
LightGCN	0.1651	0.1834	0.2613	0.0357	0.0421	0.0518	0.1919	0.4215	0.5577	0.0496	0.0785	0.0961
HGNNRec	0.1641	0.1878	0.2603	0.0361	0.0427	0.0525	0.2035	0.4372	0.5623	0.0515	0.0817	0.1013
AHGNNRec	**0.1875**	**0.2043**	**0.2747**	**0.0393**	**0.0458**	**0.0545**	**0.2257**	**0.4632**	**0.5801**	**0.0582**	**0.0869**	**0.1142**

Gain at rank K (NDCG@K) is

$$\text{NDCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}} \tag{18}$$

where DCG@K is the Discounted Cumulative Gain at rank K and IDCG@K is the Ideal Discounted Cumulative Gain at rank K. The formulas for DCG@K and IDCG@K are

$$\text{DCG@K} = \sum_{i=1}^{K} \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)} \tag{19}$$

$$\text{IDCG@K} = \sum_{i=1}^{|G|} \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)} \tag{20}$$

where rel_i is the relevance score of the ith item, $|G|$ is the total number of items in the ground truth set, and $\log_2$ is the base 2 logarithm. The formula for Hit Ratio at rank K (HR@K) is

$$\text{HR@K} = \frac{1}{|U|} \sum_{u \in U} \text{hit}(u) \tag{21}$$

where $|U|$ is the total number of users, $\text{hit}(u)$ is an indicator function that returns 1 if the recommended item is in the ground-truth set for user u up to rank K, and 0 otherwise. In other words, $\text{hit}(u) = 1$ if the user u is satisfied with the recommendation up to rank K, and 0 otherwise. To facilitate fair comparison, we set: 1) the number of GCN propagation layers; 2) the dimension of propagation layers; and 3) the dropout ratio as the optimal values of the pretrained HGNNRec.

B. Performance Comparison With Baselines

We conduct a number of experiments and the top-K recommendation performance results are illustrated in Table I. To fully evaluate the effectiveness of AHGNNRec, we set K to 10, 50, and 100, respectively, in experiment. From Table I, we summarize the following key observations.

1) In all cases, the proposed model AHGNNRec has better performance over all the baselines. We attribute the superiority of AHGNNRec to the following two properties: 1) The hierarchical propagation mechanism adopted a more comprehensive way to merge hidden semantics of different meta-paths and 2) to further enhance the model's generalization and robustness, AT is added to the recommendation training process by dynamically constructing adversarial examples for a potentially faked node feature attack.
2) Among all the baseline methods, HIG-based methods (SemRec, HERec, and CDPRec) have better performance than CF methods (LFM, BPR-FM, and AMF) and the content-based method Bi-LSTM in most cases, which shows the advantage of auxiliary heterogeneous information. The improvements of Bi-LSTM [46] method for an LFM and BPR-MF methods confirm that traditional CF models can be enhanced through the addition of rich node features. It is noteworthy that the HERec and CDPRec model works relatively well among these baselines, since this method adopts a similar path-guided random walk to generate a context sequence for user/item embeddings. However, HIG-based methods such as CDPRec show inferior performance compared to HGNNRec. CDPRec extracts a homogeneous graph from the heterogeneous graph using a meta-path-based random walker and then applies GNNs to learn user and item embeddings for recommendations. Unlike HGNNRec, CDPRec does not have a hierarchical aggregation

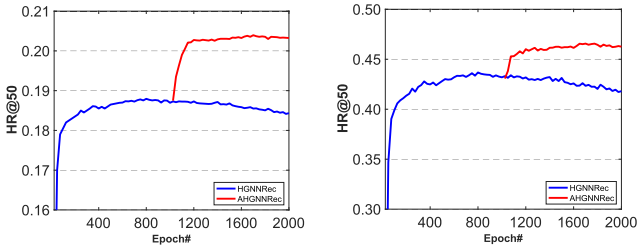


Fig. 4. Training curves of HGNNRec and AHGNNRec evaluated per 50 epochs. (a) YouTube. (b) Yelp.

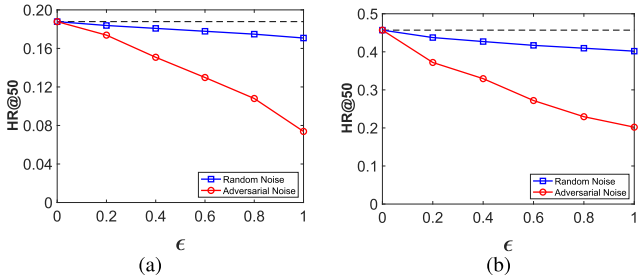


Fig. 5. Impact of adding random and adversarial perturbations to the HGNNRec recommendation model. The key observation is that adversarial perturbations have a large impact on the neural graph-based recommendation. (a) YouTube. (b) Yelp.

process that can assess the different contributions of semantic types and generate more effective user and item embeddings for recommendations.

3) Moreover, when compared to nonadversarial methods (LFM, BPR-MF, and HGNNRec), AMF and AHGNNRec outperform all of these. This demonstrates the benefit of the AT mechanism. It is worth noting that the proposed AHGNNRec achieves better performance than AMF; this demonstrates that the powerful relation dependency modeling ability of HIG, and our proposed hierarchical propagation method, provide an additional ability to exploit the hidden high-order interactions.
4) Among all the baselines, the GNN-based methods (CDPRec, NGCF, LightGCN, and HGNNRec) perform the best, which verifies that GNNs can better learn the complex relations between nodes in online social networks. HGNNRec outperforms LightGCN in most cases, since HGNNRec applies GNNs to meta-path-aware user–item graphs rather than unifying the whole graph, which can provide more fine-grained interaction information. The performance behaviors of AHGNNRec generally verify the fact that recommendation models with effective adversarial learning have the potential to generate higher recommendation accuracy.

C. Ablation and Effectiveness Analyses

To validate the effectiveness of the adversarial learning mechanism, we conduct two AT experiments investigating two aspects: generalization and robustness.

1) Generalization of Training Process: The training processes of HGNNRec and AHGNNRec are shown in Fig. 4, which are evaluated on the YouTube and Yelp datasets per 50 epochs. We first train normal HGNNRec without the AT until convergence (after about 1000 epochs). As discussed above, the model parameters of AHGNNRec are pretrained by HGNNRec; thus, we then continue training with our proposed AHGNNRec. To facilitate better comparison, we continue to train HGNNRec along with AHGNNRec. As can be observed, by employing AT mechanism to HGNNRec, we can continue to increase the performance to a great degree. By comparison, further training HGNNRec produces minimal improvement, or may even lead to a certain decline due to overfitting. For example, the best HR@50 of HGNNRec for the YouTube and Yelp datasets are 0.1878 and 0.4372, respectively, which are gradually increased to 0.2043 and 0.4632 through training by AHGNNRec. These results have proven the effeteness of the adversarial learning mechanism employed in AHGNNRec, which can make the recommendation model learn better parameters that enhance its generalization ability of the training process.

2) Robustness Against Adversarial Perturbations: To measure the model robustness in a quantitative way, we employ adversarial perturbations to the node embedding and then measure the level of performance decline, with a smaller drop ratio indicating stronger robustness [39], [40]. As shown in Fig. 5, we can find that the adversarial perturbations have a much larger impact on the graph-based recommendation model HGNNRec than random perturbations. Specifically, the random perturbation can worsen the performance of HGNNRec by just a modest ratio. By contrast, the adversarial perturbation mechanism introduced in (11) has a very dire and obvious impact on HGNNRec. For example, when ϵ is set to 1.0, the performance results in HR@50 for the YouTube dataset drops 5.85% under random perturbation but decreases 59.7% under adversarial perturbation, which is around ten times more seriously.

D. Hyperparameter Studies

We next investigate how the hyperparameters affect our model. Given a specific hyperparameter, we evaluate the model performance when adjusting its value and fixing the other hyperparameters with optimal values. Our model introduces three additional hyperparameters: 1) scale of graph adversarial perturbations (ϵ); 2) weight of the graph adversarial regularizer (α); and 3) weight of the smooth adversarial regularizer (β). Moreover, considering that the number of candidate combinations exponentially increases with the number of hyperparameters, we explore whether comparable performance could be achieved when tuning one hyperparameter alone and fixing the others with empirically determined values. It should be noted that previous works [24], [36] show that the performance can be made improved through the optimization of ϵ alone. For small values of ϵ, the hyperparameters α and β play a similar role as ϵ, where the strength of the regularization in AHGNNRec is proportional to the variation of ϵ. Thus, we fix α and β to default values of 1 and vary ϵ and summarize the performance of AHGNNRec in Fig. 6. We can observe that the performance first improves with the increase of ϵ and then drops dramatically after ϵ passes some threshold.

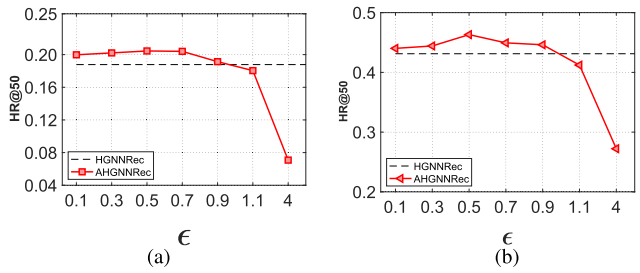


Fig. 6. Performance of AHGNNRec with respect to different values of adversarial noise level ϵ. AHGNNRec obtains the best performance when $\epsilon = 0.5$. (a) YouTube. (b) Yelp.

It suggests that an appropriate setting of ϵ improves the model robustness and generalization ability, while adversarial perturbation with an overly large norm constraint can destroy the learning process of embedding vectors. This result supports the assumption that the perturbations must be small-scale so that the constructed adversarial examples have similar feature distributions to the real data.

V. Related Work

In this section, we briefly describe related works on graph-based recommendation and graph AT.

A. Graph-Based Recommendation

With the rapid development of online social networks, we can use fruitful auxiliary data for recommendations, such as user/item attributes and social relationships. In consequence, lots of *graph-based recommendation* models have been proposed to make use of this rich context information for better recommendation results. Previous studies on graph-based recommendation methods have two main implementation strategies: path-based and embedding-based. Path-based methods exploit hidden interactions among users and items in graphs with the help of predefined meta-paths. For example, SemRec [45] exploits the hidden semantic information to learn users' different recommendation preferences. However, these HIG-based methods did not consider the multimodal content feature and neglected the crucial noise attack problem.

Graph embedding-based recommendation is another widely studied avenue, which makes great strides toward efficiently mining complex and large-scale graph structure data. In [15], [47], [48], and [49], the embedding-based method learns node embedding from graph structure recommendation data and then uses the learned user or item node embeddings to calculate the probability of prediction [50], [51], [52]. Recently, the excellent performance achieved by GNNs has provided a strong basis and opportunity for learning graph-structure data for recommendations. GNNs adopt embedding propagation to capture the dependence of graph nodes. By stacking the propagation layers, GNNs iteratively aggregate the high-order context information in the graph, enabling them to mine the fruitful hidden semantic relationship among users and items. Our method is conceptually inspired by the new GNN graph embedding methods, which provide a more principal way to handle both graph structures, such as PinSage [12], LightGCN [8], and UltraGCN [10]. Different from the previous works, the proposed AHGNNRec approach not only focuses on adding auxiliary context data to compensate for the data sparsity, but also applies hierarchical aggregation to the representation learning of users and items to improve recommendation performance. In addition, to enhance the robustness of the recommendation systems, we employ the AT mechanism in the training process.

B. Graph AT

Despite the exemplary performance of graph neural networks (GNNs) in graph-structured data representation learning, they have also been revealed to be vulnerable to small but intentional perturbation attacks on features of the input node. We further argue that the effects of perturbation attacks can do much more damage on GNNs than on standard neural networks due to the propagation process of GNNs. The cross-connected nodes would accumulate the impact of perturbations from far-away neighbor nodes that connected to the target node (i.e., the nodes to which we add perturbations attack with the aim of misleading their prediction). The propagation of information might lead to cascading effects, where manipulating a single node will affect many others [25]. AT can be seen as a simple, yet effective regularization technique, which is designed to proactively improve the model robustness by applying generated adversarial perturbations into the training process of neural networks. However, applying AT to the graph is hard due to the discrete property of graph structure data. Thus, many previous works have proposed to apply small adversarial perturbations to the continuous spaces of representation learning [24], [53], [54], [55]. GraphAT [24] attacks the graph smoothness constraint by iteratively generating adversarial examples, enabling it to improve the local smoothness of predictions over the given graph. Latent adversarial training (LAT) [54] proposes a new regularization technique for GNNs by only virtually perturbing the first embedding layer of GNNs. Free large-scale adversarial augmentation on graphs (FLAG) [55] attempts to make the model invariant to small fluctuations by iteratively adding adversarial perturbations during training. These methods are designed for semisupervised node classification, while AT should function regardless of the task at hand.

VI. Conclusion

In this article, we propose AHGNNRec, a neural framework that incorporates HIGs into recommender systems in a natural and intuitive way. AHGNNRec encodes the global node context into comprehensive node embedding to facilitate click-through rate prediction. To further improve the generalization and robustness of the HGNN-based recommendation, we apply AT to the training process by dynamically constructing adversarial examples. Experimental results on two widely used datasets demonstrate the superiority of our method over strong baselines.

References

[1] L. Wu, X. He, X. Wang, K. Zhang, and M. Wang, "A survey on accuracy-oriented neural recommendation: From collaborative filtering to information-rich recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 5, pp. 4425–4445, Jan. 2022.

[2] J. Gao, T. Zhang, and C. Xu, "A unified personalized video recommendation via dynamic recurrent neural networks," in *Proc. 25th ACM Int. Conf. Multimedia*, Oct. 2017, pp. 127–135.

[3] G. Jain, T. Mahara, S. C. Sharma, and A. K. Sangaiah, "A cognitive similarity-based measure to enhance the performance of collaborative filtering-based recommendation system," *IEEE Trans. Computat. Social Syst.*, vol. 9, no. 9, pp. 1–9, Jul. 2022.

[4] Y. Xu et al., "Machine learning-driven APPs recommendation for energy optimization in green communication and networking for connected and autonomous vehicles," *IEEE Trans. Green Commun. Netw.*, vol. 6, no. 3, pp. 1543–1552, Sep. 2022.

[5] H. Gao, X. Qin, R. J. D. Barroso, W. Hussain, Y. Xu, and Y. Yin, "Collaborative learning-based industrial IoT API recommendation for software-defined devices: The implicit knowledge discovery perspective," *IEEE Trans. Emerg. Topics Computat. Intell.*, vol. 6, no. 1, pp. 66–76, Feb. 2022.

[6] Y. Xu, L. Zhu, Z. Cheng, J. Li, and J. Sun, "Multi-feature discrete collaborative filtering for fast cold-start recommendation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 1, 2020, pp. 270–278.

[7] L. Sang, M. Xu, S. Qian, M. Martin, P. Li, and X. Wu, "Context-dependent propagating-based video recommendation in multimodal heterogeneous information networks," *IEEE Trans. Multimedia*, vol. 23, pp. 2019–2032, 2021.

[8] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 639–648.

[9] R. Zhang, S. Yang, Q. Zhang, L. Xu, Y. He, and F. Zhang, "Graph-based few-shot learning with transformed feature propagation and optimal class allocation," *Neurocomputing*, vol. 470, pp. 247–256, Jan. 2022.

[10] K. Mao, J. Zhu, X. Xiao, B. Lu, Z. Wang, and X. He, "UltraGCN: Ultra simplification of graph convolutional networks for recommendation," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 1253–1262.

[11] Z. Yang, Y. Wang, Y. Cheng, T. Zhang, and X. Wang, "Recommendation model based on enhanced graph convolution that fuses review properties," *IEEE Trans. Computat. Social Syst.*, early access, Jun. 14, 2022, doi: 10.1109/TCSS.2022.3181065.

[12] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 974–983.

[13] Y. Wei, X. Wang, L. Nie, X. He, R. Hong, and T.-S. Chua, "MMGCN: Multi-modal graph convolution network for personalized recommendation of micro-video," in *Proc. 27th ACM Int. Conf. Multimedia*, Oct. 2019, pp. 1437–1445.

[14] J. Liu, C. Shi, C. Yang, Z. Lu, and P. S. Yu, "A survey on heterogeneous information network based recommender systems: Concepts, methods, applications and resources," *AI Open*, vol. 3, pp. 40–57, Jan. 2022.

[15] C. Shi, B. Hu, W. X. Zhao, and P. S. Yu, "Heterogeneous information network embedding for recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 2, pp. 357–370, Feb. 2019.

[16] C. Shi, X. Wang, and P. S. Yu, *Heterogeneous Graph Representation Learning and Applications* (Artificial Intelligence: Foundations, Theory, and Algorithms). Singapore: Springer, 2022, doi: 10.1007/978-981-16-6166-2.

[17] L. Sang, M. Xu, S. Qian, and X. Wu, "Knowledge graph enhanced neural collaborative recommendation," *Exp. Syst. Appl.*, vol. 164, Feb. 2021, Art. no. 113992.

[18] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2019, pp. 793–803.

[19] L. Sang, M. Xu, S. Qian, and X. Wu, "Multi-modal multi-view Bayesian semantic embedding for community question answering," *Neurocomputing*, vol. 334, pp. 44–58, Mar. 2019.

[20] W. Jin et al., "Adversarial attacks and defenses on graphs," *ACM SIGKDD Explor. Newslett.*, vol. 22, no. 2, pp. 19–34, 2021.

[21] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 2847–2856.

[22] F. Zhang and S. Wang, "Detecting group shilling attacks in online recommender systems based on bisecting K-means clustering," *IEEE Trans. Computat. Social Syst.*, vol. 7, no. 5, pp. 1189–1199, Oct. 2020.

[23] F. Wang, H. Zhu, G. Srivastava, S. Li, M. R. Khosravi, and L. Qi, "Robust collaborative filtering recommendation with user-item-trust records," *IEEE Trans. Computat. Social Syst.*, vol. 9, no. 4, pp. 986–996, Aug. 2021.

[24] F. Feng, X. He, J. Tang, and T.-S. Chua, "Graph adversarial training: Dynamically regularizing based on graph structure," 2019, *arXiv:1902.08226*.

[25] M. Fang, G. Yang, N. Z. Gong, and J. Liu, "Poisoning attacks to graph-based recommender systems," in *Proc. 34th Annu. Comput. Secur. Appl. Conf.*, Dec. 2018, pp. 381–392.

[26] S. Zhang, H. Yin, T. Chen, Q. V. N. Hung, Z. Huang, and L. Cui, "GCN-based user representation learning for unifying robust recommendation and fraudster detection," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 689–698.

[27] P.-A. Chirita, W. Nejdl, and C. Zamfir, "Preventing shilling attacks in online recommender systems," in *Proc. 7th Annu. ACM Int. Workshop Web Inf. Data Manage.*, Nov. 2005, pp. 67–74.

[28] M. Si and Q. Li, "Shilling attacks against collaborative recommender systems: A review," *Artif. Intell. Rev.*, vol. 53, no. 1, pp. 291–319, Jan. 2020.

[29] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–11.

[30] W. Jin et al., "Adversarial attacks and defenses on graphs: A review, a tool and empirical studies," 2020, *arXiv:2003.00653*.

[31] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2019, pp. 165–174.

[32] L. Sang, M. Xu, S. Qian, and X. Wu, "AAANE: Attention-based adversarial autoencoder for multi-scale network embedding," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining (PAKDD)*, 2019, pp. 3–14.

[33] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 1024–1034.

[34] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.

[35] J. Chen, X. Lin, Z. Shi, and Y. Liu, "Link prediction adversarial attack via iterative gradient attack," *Trans. Comput. Social Syst.*, vol. 7, no. 4, pp. 1081–1094, 2020.

[36] T. Miyato, S.-I. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: A regularization method for supervised and semi-supervised learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 1979–1993, Aug. 2019.

[37] H. Chen, K. Zhou, K.-H. Lai, X. Hu, F. Wang, and H. Yang, "Adversarial graph perturbations for recommendations at scale," in *Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2022, pp. 1854–1858.

[38] J. Chen, X. Lin, H. Xiong, Y. Wu, H. Zheng, and Q. Xuan, "Smoothing adversarial training for GNN," *IEEE Trans. Computat. Social Syst.*, vol. 8, no. 3, pp. 618–629, Jun. 2021.

[39] J. Tang, X. Du, X. He, F. Yuan, Q. Tian, and T.-S. Chua, "Adversarial training towards robust multimedia recommender system," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 5, pp. 855–867, May 2020.

[40] X. He, Z. He, X. Du, and T.-S. Chua, "Adversarial personalized ranking for recommendation," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jun. 2018, pp. 355–364.

[41] M. Yan, J. Sang, and C. Xu, "Mining cross-network association for YouTube video promotion," in *Proc. 22nd ACM Int. Conf. Multimedia*, Nov. 2014, pp. 557–566.

[42] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2008, pp. 426–434.

[43] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. 25th Conf. Uncertainty Artif. Intell. (AUAI)*. Montreal, QC, Canada: AUAI Press, 2009, pp. 452–461.

[44] T. Xie, Y. Xu, L. Chen, Y. Liu, and Z. Zheng, "Sequential recommendation on dynamic heterogeneous information network," in *Proc. IEEE 37th Int. Conf. Data Eng. (ICDE)*, Apr. 2021, pp. 2105–2110.

[45] C. Shi, Z. Zhang, P. Luo, P. S. Yu, Y. Yue, and B. Wu, "Semantic path based personalized recommendation on weighted heterogeneous information networks," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2015, pp. 453–462.

[46] Y. Zhang et al., "Sequential click prediction for sponsored search with recurrent neural networks," in *Proc. AAAI*, vol. 14, 2014, pp. 1369–1375.

[47] H. Wang, F. Zhang, X. Xie, and M. Guo, "DKN: Deep knowledge-aware network for news recommendation," in *Proc. World Wide Web Conf. World Wide Web (WWW)*, 2018, pp. 1835–1844.
[48] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 353–362.
[49] Y. Yin, Y. Li, H. Gao, T. Liang, and Q. Pan, "FGC: GCN based federated learning approach for trust industrial service recommendation," *IEEE Trans. Ind. Informat.*, vol. 19, no. 3, pp. 3240–3250, Oct. 2022.
[50] C. Gao, X. Wang, X. He, and Y. Li, "Graph neural networks for recommender system," in *Proc. 15th ACM Int. Conf. Web Search Data Mining*, Feb. 2022, pp. 1623–1625.
[51] K. Liu, F. Xue, X. He, D. Guo, and R. Hong, "Joint multi-grained popularity-aware graph convolution collaborative filtering for recommendation," *IEEE Trans. Computat. Social Syst.*, vol. 10, no. 1, pp. 72–83, Feb. 2022.
[52] L. Sang, M. Xu, S. Qian, and X. Wu, "Knowledge graph enhanced neural collaborative filtering with residual recurrent network," *Neurocomputing*, vol. 454, pp. 417–429, Sep. 2021.
[53] Z. Deng, Y. Dong, and J. Zhu, "Batch virtual adversarial training for graph convolutional networks," 2019, *arXiv:1902.09192*.
[54] H. Jin and X. Zhang, "Latent adversarial training of graph convolution networks," in *Proc. ICML Workshop Learn. Reasoning Graph-Structured Represent.*, vol. 2, 2019, pp. 1–7.
[55] K. Kong et al., "Robust optimization as data augmentation for large-scale graphs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 60–69.

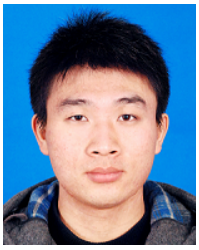
Lei Sang received the Ph.D. degree from the Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, Australia, in 2021.

He is currently a Lecturer with the School of Computer Science and Technology, Anhui University, Anhui, China. His current research interests include natural language processing, data mining, and recommendation systems.

Min Xu (Member, IEEE) received the B.E. degree from the University of Science and Technology of China, Hefei, China, the M.S. degree from the National University of Singapore, Singapore, and the Ph.D. degree from the University of Newcastle, Callaghan, Australia.

She is an associate professor with the School of Electrical and Data Engineering (SEDE), Faculty of Engineering and Information Technology (FEIT), University of Technology Sydney (UTS), and the Leader of Visual and Aural Intelligence Laboratory with the Global Big Data Technologies Centre (GBDTC), UTS. She has published 180+ research papers in prestigious international journals and conference proceedings, including T-PAMI, T-NNLS, T-MM, T-MC, PR, ICLR, ICML, CVPR, ICCV, ACM MM, AAAI. Her research interests include multimedia, computer vision, and machine learning.

Dr. Xu is an Editorial Board member for *Elsevier Journal of Neurocomputing and Journal of Ambient intelligence and Humanised Computing*; and served as a Program Chair/Area Chair for many major conferences.



Shengsheng Qian (Member, IEEE) received the B.E. degree from Jilin University, Changchun, China, in 2012, and the Ph.D. degree in pattern recognition and intelligent systems from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2017.

He is currently an Associate Professor with the Institute of Automation, Chinese Academy of Sciences. His current research interests include social media data mining and social event content analysis.

Xindong Wu (Fellow, IEEE) received the Ph.D. degree in artificial intelligence from The University of Edinburgh, Edinburgh, U.K., in 1993.

He is currently a Professor with the Hefei University of Technology, Hefei, China. His current research interests include data mining, knowledge-based systems, and Web information exploration.

Prof. Wu is a fellow of AAAS. He is the Steering Committee Chair of the IEEE International Conference on Data Mining (ICDM). He is also the Editor-in-Chief of Knowledge and Information Systems (KAIS) and ACM Transactions on Knowledge Discovery from Data (TKDD).