MGPolicy: Meta Graph Enhanced Off-policy Learning for Recommendations

Xiangmeng Wang
xiangmeng.wang@student.uts.edu.au
University of Technology Sydney
Sydney, Australia

Qian Li[†][*]
qli@curtin.edu.au
Curtin University
Perth, Australia

Dianer Yu
Dianer.Yu-1@student.uts.edu.au
University of Technology Sydney
Sydney, Australia

Zhichao Wang
zchaoking@gmail.com
University of New South Wales
Sydney, Australia

Hongxu Chen
Hongxu.Chen@uts.edu.au
University of Technology Sydney
Sydney, Australia

Guandong Xu[†]
guandong.xu@uts.edu.au
University of Technology Sydney
Sydney, Australia

ABSTRACT

Off-policy learning has drawn huge attention in recommender systems (RS), which provides an opportunity for reinforcement learning to abandon the expensive online training. However, off-policy learning from logged data suffers biases caused by the policy shift between the target policy and the logging policy. Consequently, most off-policy learning resorts to inverse propensity scoring (IPS) which however tends to be over-fitted over exposed (or recommended) items and thus fails to explore unexposed items.

In this paper, we propose meta graph enhanced off-policy learning (MGPolicy), which is the first recommendation model for correcting the off-policy bias via contextual information. In particular, we explicitly leverage rich semantics in meta graphs for user state representation, and then train the candidate generation model to promote an efficient search in the action space. Moreover, our MGpolicy is designed with counterfactual risk minimization, which can correct policy learning bias and ultimately yield an effective target policy to maximize the long-run rewards for the recommendation. We extensively evaluate our method through a series of simulations and large-scale real-world datasets, achieving favorable results compared with state-of-the-art methods. Our code is currently available at https://www.dropbox.com/sh/9ugr1lx7gzwfub4/AABY46hVG6qKJnGAWjRJZMFKa?dl=0

CCS CONCEPTS

• **Computer systems organization → Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

KEYWORDS

Recommendation; Off-policy Learning; Counterfactual Risk Minimization; Bias

ACM Reference Format:
Xiangmeng Wang, Qian Li, Dianer Yu, Zhichao Wang, Hongxu Chen, and Guandong Xu. 2022. MGPolicy: Meta Graph Enhanced Off-policy Learning for Recommendations. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3477495.3532021

1 INTRODUCTION

Recommender system (RS) has become prevalent in Web applications to help users seek preferred content from massive information provided [4, 19]. Traditional RS including collaborative filtering [20] and knowledge-based systems [11] treat the recommendation as a static process following a fixed greedy strategy [20, 22]. Traditional RS are static and can not adapt to the sequential nature of user interaction with the system. Recently, Reinforcement Learning (RL) that learns the optimal target recommendation policy to maximize long-term user satisfaction has drawn huge attention in RS [2]. Particularly, RL-based recommendation trains an agent (recommender) via online learning from real-time user interaction trajectories. However, such online learning is infeasible in real RS since it might harm user satisfaction and deteriorate the revenue of the platform [1, 9]. Fortunately, off-policy learning emerges as a favorable opportunity for policy optimization, which uses logged user feedback instead of constructing expensive online interactive environments [14, 24, 38].

To abandon the online training, as shown in Figure 1, the off-policy learning needs to find an optimal target policy π_θ that maximizes users' long-term satisfactions by given logged data collected by the logging policy π_0. Thus, the off-policy learning has to fundamentally address the counterfactual question: *what the cumulative reward (i.e., users' feedback during a period) would be if a new target policy had been deployed instead of the original logging policy* [32, 33]. Nevertheless, using the logged feedback data for answering this counterfactual question is not easy, since the target policy is different from the historical logging policy in the off-policy setting [23, 26, 30, 31]. As shown in Figure 1, the two policies hold different distributions while rare actions chosen by the target policy

[*] Contributing equally with the first author.
[†] Corresponding author.

never appear in the original logging policy. Recent off-policy learning leverages inverse propensity score (IPS) [3, 23] to correct the bias caused by the policy shift. These methods re-weight each sample via the propensity score, i.e., the probability ratio between the target policy and the behavior policy, to get an unbiased empirical risk minimization objective over the logged data.

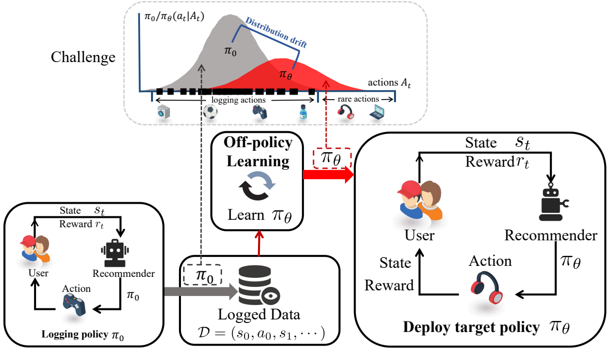


Figure 1: Off-policy learning for recommendation.

However, a certain amount of unexposed/unrecommended items do not have any feedback (reward), which renders IPS to be over-fitted and thus degrades the off-policy performance. For example, as shown in Figure 1, compared with a large action space (e.g., items) in a recommender system, actions taken by users are limited in a deficiency action space due to the existence of biases (e.g., exposure bias or selection bias) [21]. In such case, IPS tends to be over-fitted to exposed/recommended items (actions), i.e., an earphone will not be nominated in the target policy simply because it has never appeared in the behavior logging policy, leading to the "poor gets poorer" phenomenon [7].

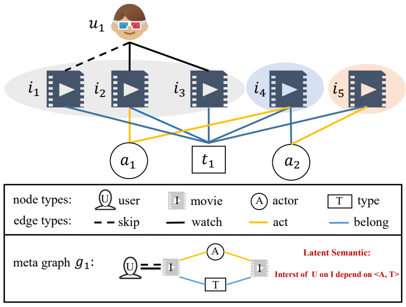


Figure 2: Toy example of inferring missing rewards from meta graphs.

Fortunately, there exists rich real-world prior context information available outside the recommender systems that are useful for inferring the missing rewards of rare actions. We argue that the higher-order dependencies of user/action attributes underlying the meta graph [13] can deal with incomplete log data and empower off-policy learning in the recommendation. Take an example shown in Figure 2, we have a user u_1 interacts with (i_1, i_2, i_3) but not with (i_4, i_5). As a result, it is impossible to learn cumulative rewards from u_1 for a target policy that selects actions not included in the logged data (i.e., i_4, i_5). However, we can infer the rewards of selecting i_4 and i_5 with the assistance of the meta graph g_1 given in Figure 2. Since u_1 gives positive feedback to (i_2, i_3), we can infer that the user's interest is related to both a_1 and t_1. Hence, the unobserved action i_4 with the same attributes a_1 and t_1 would be very likely preferred by u_1, thus can receive a positive reward (e.g., click) from u_1. Moreover, since u_1 gives negative feedback (e.g., not click) to i_1 with a_2 and t_1, i_5 possessing the same attributes would potentially receive negative rewards from u_1 as well. To this end, the positive/negative rewards of unexposed actions can be further inferred by the context information learned from meta graphs.

In this work, we investigate how to correct off-policy biases with the assistance of meta graphs and propose a novel end-to-end off-policy learning framework, i.e., *Meta graph enhanced off-policy learning for recommendations (MGpolicy)*. To tackle the missing reward information in policy learning, we design a novel meta graph-based state representation learning that leverages the complex semantics to aggregate context-aware actions for estimating high-quality target policy. To handle the policy shift, we resort the Counterfactual Risk Minimization (CRM) [31]. Unlike re-weighting samples in conventional IPS-based correction methods, CRM directly minimizes an empirical risk estimated from the logged feedback data as it came from the true risk of the target policy, therefore fundamentally removing the distribution mismatch between target policy and logging policy. The contributions of our work can be summarized as follows.

- To the best of our knowledge, we are the first to leverage the contextual information in meta graphs and thus provide high-quality target policy learning for correcting the bias in off-policy recommendations.
- Our MGpolicy is designed with an efficient two-stage off-policy gradient method including state representation stage and candidate selection stage, which explicitly leverages rich semantics in meta graphs for user state representation and trains the candidate generation model to improve the efficiency of action search.
- Based on the context-aware states representation, a counterfactual risk minimization is designed for our MGpolicy to achieve an unbiased policy that maximizes the long-run rewards for the recommendation.
- Empirically, we generate an online environment via simulators to carry out experiments on offline datasets. Extensive results show that our methods outperform the state-of-the-art methods.

2 RELATED WORK

This work is closely related to two topics: off-policy recommendation and conventional HIN-enhanced recommendation.

2.1 Off-policy Learning for Recommendation

Reinforcement learning in recommendation usually abandons the expensive online learning while trained with the logged feedback data [1, 15, 23], i.e., the off-policy learning. The logging policy for collecting the logged data share different distribution with the target policy, thus, many works rely on off-policy bias corrections to pursue an unbiased optimization. The first category of off-policy learning is policy-based learning. Among them, POEM [31] uses inverse propensity scoring (IPS) to correct the bias while BanditNet [15] includes an additional self-normalization IPS (SNIPS). A major limitation of these methods is that IPS is likely to be overfitted as rare actions have zero probability of being taken in RS, leading to the "poor gets poorer" phenomenon [7].

Another category to off-policy learning refers to value-based learning, such as the Q-learning [44] that generates target policy by approximating Q-values of actions. Many value-based methods correct off-policy biases by mimicking data distribution of the logging policy, such as the bootstrapping error accumulation reduction (BEAR) [18]. However, as actions in value-based learning are selected by the maximum value, the policy may be disrupted badly by a single erroneously optimistic estimate, i.e., a phenomenon known as optimizer's curse [29]. Moreover, these methods focus on mimicking the distribution as it comes from a true (i.e., target) policy, which however ignores to pursue the causal-effect of the bias pattern. On the contrary, our MGpolicy takes the distribution shift as a causally counterfactual question, and resorts policy-based method augmented by the context information for the unbiased high-quality target policy learning.

2.2 Traditional HIN-based Recommendation

Heterogeneous information network (HIN) has shown its power in modeling heterogeneous sources (e.g., social relations, text reviews), thus has been wildly adopted in traditional recommenders [11, 28, 35]. Based on meta relations and meta paths of HINs, representative methods such as HGT [12], HAN [34] and MCrec [11] promisingly improve recommendations. Recently, Huang et al. propose a novel meta graph [13] to go beyond the simple chain relations of meta paths and model higher-order semantics of HINs. Later works using meta graphs have achieved significant results in recommendation tasks, e.g., FMG [43], AMERec [39]. Despite the efforts, they focus on the one-step static recommendation task, which fails to model dynamic users interactions. We consider using complex semantics in meta graphs to assist the off-policy learning under the interactive setting, which is however not been well studied.

3 PRELIMINARY

3.1 Off-policy Learning for Recommendation

The off-policy reinforcement learning problem can be defined as a data-driven formulation of the reinforcement learning problem. The recommender agent no longer dynamically interacts with users (i.e., environment) in an online manner. Instead, the off-policy learning is provided with a static logged dataset $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$ generated by an unknown policy π_0 (a.k.a logging policy). In the recommendation setting, the logging policy $\pi_0(a|s)$ selects an action $a \in \mathcal{A}$ to take (e.g., videos to recommend) conditioning on the user state $s \in \mathcal{S}$, and the recommendation receives user feedback reward $r(s, a) \in \mathcal{R}$ (i.e., clicks or watch time) for this particular state-action pair. The process can be formulated with a Markov Decision Process (MDP), where

- $\mathcal{S}$: a continuous state space describing the user states, e.g., user's contextual information involved during interactions;
- $\mathcal{A}$: a discrete action space containing items available for recommendation;
- $\mathcal{P}$: the state transition probability;
- $\mathcal{R}$: $r(s, a) \in \mathcal{R}$ is the immediate reward produced by taking the action a at the user state s;
- γ: a discount factor $\gamma \in [0, 1]$ used for future immediate rewards;

Using such a logged dataset of trajectories, the recommender aims to seek a policy π_θ that maximizes the expected cumulative rewards $R(\pi_\theta)$ over potentially infinite time horizon T, with

$$R(\pi_\theta) = \mathbb{E}_{s_0 \sim \rho(s), a_t \sim \pi_\theta(a|s_t), s_{t+1} \sim P(s|s_t, a_t)} \left[\sum_{t=0}^{T} \gamma^t r\left(s_t, a_t\right) \right] \quad (1)$$

where $\rho(s)$ is the initial distribution of user states, $P\left(s \mid s_t, a_t\right) \in \mathcal{P}$ is the state transition probability.

3.2 Meta graph

Definition 1 (Meta graph). *Given a Heterogeneous Information Network (HIN) $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with network scheme $\mathcal{T} = \{\mathcal{K}, \mathcal{J}\}$, where $\mathcal{V}$ and $\mathcal{E}$ are node and edge sets, $\mathcal{K}$ and $\mathcal{J}$ are node- and edge-type sets, respectively. A meta graph $\mathcal{M} = (\mathcal{K}^*, \mathcal{J}^*, n_s, n_t)$ is a sub-graph of $\mathcal{T}$ whose node $v \in \mathcal{K}^* \in \mathcal{K}$ and edge $e \in \mathcal{J}^* \in \mathcal{J}$. The meta graph links a single source node n_s and a single sink node n_t, while each node v (except source and target nodes) is allowed to have in- and out-degree larger than 1 and $|\mathcal{K}^*| + |\mathcal{J}^*| > 2$ (i.e., heterogeneity).*

Definition 2 (Meta graph Instance). *Given meta graph $\mathcal{M} = (\mathcal{K}^*, \mathcal{J}^*, n_s, n_t)$, a meta graph instance $g = \left(\mathcal{V}_\mathcal{M}, \mathcal{E}_\mathcal{M}\right)$ is a sub-graph of the HIN $\mathcal{G}$ whose node $v \in \mathcal{V}_\mathcal{M} \in \mathcal{V}$ and edge $e \in \mathcal{E}_\mathcal{M} \in \mathcal{E}$. Each node v and edge e in g correspond to one particular type $\phi(v) \in \mathcal{K}^*$ and $\psi(e) \in \mathcal{J}^*$ with a node type mapping function $\phi : \mathcal{V}_\mathcal{M} \rightarrow \mathcal{K}^*$ and a edge type mapping function $\psi : \mathcal{E}_\mathcal{M} \rightarrow \mathcal{J}^*$.*

3.3 Task Formulation

Previous off-policy learning [1, 2, 23] has frequently assumed the choosing of presented actions depends only on user's descriptions (e.g., user id) and its historical interactions with items, neglecting the large volume of unobserved user and item attributes.

Given the logged trajectories and meta graphs, we aim to use meta graphs in both the context-aware state representation and the candidate selection. The meta graphs record contextual information about users and actions, e.g., users' social relations or actions (items)' genres, thus is useful for inferring missing user feedback on unobserved actions in the logged data. In our setting, the contextual information is incorporated into each state s_t to scale the logged data contribution for unbiased target policy learning. Moreover, the structural correlations in meta graphs are used to guide better candidate item retrieval to address the sample efficiency issue.

4 METHODOLOGY

In this section, we provide the general off-policy learning framework (cf. Figure 3) that leverages contextual information in meta graphs for bias correction in the recommendation task.

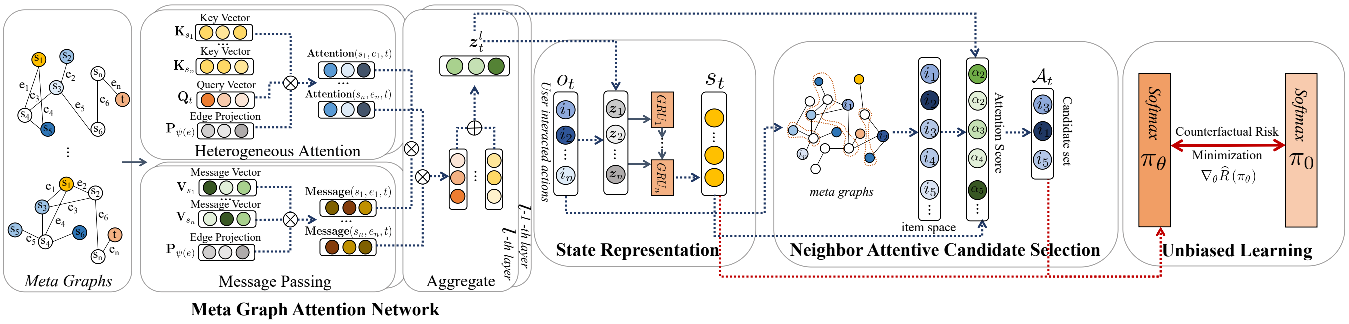


Figure 3: Our model framework of MGpolicy. Given sampled meta graphs with target node t (i.e., an item) and all its source nodes (e.g, s_1, s_n). The meta graph attention Network maps all source nodes, target node and source nodes messages as *key*, *query* and *message* vectors to learn the contextual node representation z_t^l. The State Representation takes contextual representations of items in users' interaction history o_t as inputs to recursively learn the user state s_t. The Neighbor Attentive Candidate Selection filters items into candidate set $\mathcal{A}_t$ based on item node neighbors in meta graphs and their attention scores. MGpolicy finally passes state s_t and actions $a_t \in \mathcal{A}_t$ to a counterfactual risk minimization to learn the unbiased target policy π_θ.

4.1 Meta Graph Attention Network for State Representation

We firstly present our Meta Graph Attention (MGA) network that parameterizes item entities among meta graphs as vector representations. Then we go to present the state representation learning that aggregates item representations to produce state vectors for the latter policy learning.

4.1.1 Meta Graph Attention for Node Representation. The L-layer meta graph attention network is built upon the architecture of graph Transformer [25] to recursively propagates the information from all neighbor nodes of a target node (e.g., nodes with types been *Category*, *Year*) to get the contextualized representation for the target node (i.e., an item), in which three discriminate operators, namely **Attention**, **Message passing** and **Aggregate** are performed for each layer l. The three operators rely on the node pairs $e = (s, t)$ and their type triplets $\langle \phi(s), \psi(e), \phi(t) \rangle$ among the sampled meta graph instance [1] to reveal semantic connections and the heterogeneity, where the source node s is one neighbor node of the target node t and e is the connection edge. We firstly initialize node representations at the 0-th layer of MGA with Multi-OneHot [41] by mapping node IDs into node embeddings, where the node embedding for i is denoted by z_i^0. With the initialized embeddings and node- and edge-type specific node pairs $e = (s, t)$, below we elaborate on the three operators at the l-th layer.

Attention. The attention operator parameterizes the weight matrices of the semantic connection $e = (s, t)$ with a source node projection, an edge projection, and a target node projection. Formally, we map the source node s into a *key* vector $\mathbf{K}_s$ and the target node t into a *query* vector $\mathbf{Q}_t$ which restricted to their node types $\phi(s)$ and $\phi(t)$. Moreover, unlike vanilla Transformer [25], that directly dots *key* and *query* vectors as attention vectors, we additionally dot the importance of the edge type between s and t with an edge type projection $\mathbf{P}_{\psi(e)}$ to capture diverse edge heterogeneity. Then, the weight matrices $\boldsymbol{W}^i(s, e, t)$ of $e = (s, t)$ in the i-th head of the h head attention is calculated by:

$$\mathbf{K}_s^i = \mathbf{L}_{\phi(s)}^i(z_s^{l-1}), \quad \mathbf{Q}_t^i = \mathbf{L}_{\phi(t)}^i(z_t^{l-1}),$$
$$\boldsymbol{W}^i(s, e, t) = \left(\mathbf{K}_s^i \mathbf{P}_{\psi(e)} \mathbf{Q}_t^{i\,T}\right) \cdot \frac{\mu_{\langle \phi(s), \psi(e), \phi(t) \rangle}}{\sqrt{d}} \tag{2}$$

where $\mathbf{K}_s^i$ is the *key* vector in the i-th head derived by the linear projection $\mathbf{L}_{\phi(s)}^i : \mathbb{R}^d \rightarrow \mathbb{R}^{\frac{d}{h}}$, where h is the number of attention heads. Note that $\mathbf{L}_{\phi(s)}^i$ is specified to the node type $\phi(s)$, such that each type of nodes has a unique linear projection to model the distribution of type difference. Similarly, the *query* vector $\mathbf{Q}_t^i$ is also derived by the linear projection specified to the target node type $\phi(t)$. The μ is the prior tensor that reveals the significance of each type triplet $\langle \phi(s), \psi(e), \phi(t) \rangle$, serving as an adaptive scaling to the attention.

Finally, we concatenate all h attention heads to get the attention vector for each node pair $e = (s, t)$ with:

$$\textbf{Attention}(s, e, t) = \text{Softmax}_{\forall s \in \mathcal{N}(t)} \left(\|_{i=1}^{h} \boldsymbol{W}^i(s, e, t) \right) \tag{3}$$

where $\|$ is the concatenation operation and $\mathcal{N}(t)$ is the target node neighbors set of t, i.e, all source nodes that connect with t within meta graphs.

Message passing. The message passing operator calculates the messages of source nodes s that wait to be propagated to the target node t. Formally, at the i-th head of message passing, we firstly project $\phi(s)$-type source node s into the i-th *message* vector $\mathbf{V}_s^i$, then product the message vector with the edge type projection $\mathbf{P}_{\psi(e)}$ to further capture the importance of edge type $\psi(e)$.

$$\mathbf{V}_s^i = \mathbf{L}_{\phi(s)}^i(z_s^{l-1})$$
$$\boldsymbol{M}^i(s, e, t) = \mathbf{V}_s^i \mathbf{P}_{\psi(e)} \tag{4}$$

where $\mathbf{V}_s^i$ is derived by the linear projection $\mathbf{L}_{\phi(s)}^i : \mathbb{R}^d \rightarrow \mathbb{R}^{\frac{d}{h}}$, serving as the *message* vector at the i-th head of message passing.

Then we concatenate all h message heads to get the contextual information from source node s to target node t:

$$\textbf{Message}(s, e, t) = \|_{i=1}^{h} \boldsymbol{M}^i(s, e, t) \tag{5}$$

[1] The meta graph instances are extracted by the meta graph sampling [27].

Aggregate. With the attention weights and messages from source nodes, the aggregate operator weights sum messages from source nodes (i.e., target node neighbors) to learn the attentive representation of the target node. Formally, the attention score $\mathbf{Attention}(s, e, t)$ and the message $\mathbf{Message}(s, e, t)$ are producted as weighted representation of node pair $e = (s, t)$, then the representations of all node pairs that connect with t are aggregated into z_t^l, serving as the contextual representation of t at layer l:

$$z_t^l = \bigoplus_{\forall s \in \mathcal{N}(t)} (\mathbf{Attention}(s, e, t) \cdot \mathbf{Message}(s, e, t)) \tag{6}$$

where $\oplus$ is the aggregation operator and $\cdot$ is the element-wise product. $\mathcal{N}(t)$ is the node neighbors set of the target node t (i.e., an item).

High-order propagation By stacking L layers of our MGA, we can get the final node representation of the target node t (i.e., an item). We adopt layer-aggregation mechanism [40] to concatenate the representations at each layer into a single vector, as follows:

$$z_t = z_t^1 + \cdots + z_t^L \tag{7}$$

By doing so, we can capture higher-order propagations of node pairs across different MGA layers. Finally, the z_t is outputted by our MGA, serving as the contextual representation of an item t for the next state representation learning.

4.1.2 Meta graph-based State Representation. We now present the state representation learning that aggregates contextual item representations into each user state representation s_t. Generally, the state representations s_t are extracted from actions (i.e., items) $o_t = \{i_1, i_2, ..., i_n\}$ taken at each time t along the trajectory. Since we have learned the representation of each item through our MGA, we can leverage contextual item representations for better state representation learning. Formally, for a state s_t at time step t, we have a set of items/actions $o_t = \{i_1, i_2, ..., i_n\}$ recommended/taken by the recommender agent. The representation of each item i has been learned through Eq. (7), which is denoted by z_i. Considering o_t have sequential patterns [1], we resort Recurrent Neural Networks (RNN) with a gated recurrent unit (GRU) [8] to capture the sequential information in user interaction trajectory. We firstly initialize the state representation s_0 with an initial state distribution $s_0 \sim \rho_0$ [2]. Then we learn state representation s_t through the recurrent cell by aggregating representations of items in o_t:

$$\begin{aligned} \mathbf{u}_t &= \sigma_g \left(\mathbf{W}_1 z_t + \mathbf{U}_1 s_{t-1} + \mathbf{b}_1\right) \\ \mathbf{r}_t &= \sigma_g \left(\mathbf{W}_2 z_t + \mathbf{U}_2 s_{t-1} + \mathbf{b}_2\right) \\ \hat{s}_t &= \sigma_h \left(\mathbf{W}_3 z_t + \mathbf{U}_3 \left(\mathbf{r}_t \cdot s_{t-1}\right) + \mathbf{b}_3\right) \\ s_t &= (1 - \mathbf{u}_t) \cdot s_{t-1} + \mathbf{u}_t \cdot \hat{s}_t \end{aligned} \tag{8}$$

where $\mathbf{u}_t$ and $\mathbf{r}_t$ denote the update gate and reset gate vector generated by GRU, $\cdot$ is the element-wise product operator, $\mathbf{W}_i$, $\mathbf{U}_i$ are weight matrix and $\mathbf{b}_i$ are the bias vectors. Particularly, the hidden state s_t is generated by a GRU with inputs of a previous hidden state s_{t-1} and a new candidate hidden state $\hat{s}_t$. Finally, s_t serves as the state representation at time step t.

4.2 Neighbor Attentive Candidate Selection

Due to the large action space, calculating rewards for all actions with limited user interaction data would lead to a well-known sample efficiency issue [28, 45]. Thus, it is highly desirable to develop a candidate selection strategy that filters out irrelevant items to improve the efficiency of the action search. Previous method [45] uniformly selects multi-hop neighbors of users' interacted items among a Knowledge Graph (KG) as candidates. However, interacted items can connect with tremendous neighbors in the KG and their number exponentially grow within higher-hop connections. As a result, this manner would lead to a large candidate action space, degrading the policy learning ultimately. Moreover, different neighbors connected by items contribute differently to the user state, which thus should be accounted for differently in the candidates selection. To alleviate the sample efficiency issue effectively, we propose a neighbor attentive candidate selection strategy that samples meta graph neighbors of users' interacted items based on their importance to the current user state.

Formally, having obtained the item node representation z_i for an item i from Eq. (7), and the user state representation s_t from Eq. (8). We firstly implement the attention mechanism based on s_t and z_i to calculate the attention score α_i as follows:

$$\alpha_i = \mathrm{ReLU}\left(\boldsymbol{W}_s s_t + \boldsymbol{W}_z z_i + \mathbf{b}\right) \tag{9}$$

where $\boldsymbol{W}_s$, $\boldsymbol{W}_z$ are the two weight matrices and $\mathbf{b}$ is the bias vector.

Hereafter, we normalize the attentive scores across all meta graph-based neighbors connected with item i by the softmax function:

$$\hat{\alpha_i} = \frac{\exp\left(\alpha_i\right)}{\sum_{i \in \mathcal{N}_{(i)}} \exp\left(\alpha_i'\right)}. \tag{10}$$

where $\mathcal{N}_{(i)}$ is the meta graph-based neighbors set of item i. As a result, the final attention score $\hat{\alpha_i}$ is capable of indicating which neighbor item should contribute more to the current user state.

We then select n-top neighbors of users' interacted items (i.e., o_t) into the candidate set $\mathcal{A}_t$ indicated by their attention scores.

$$\mathcal{A}_t = \left\{ i \mid i \in \bigcup_{i=1}^{n} \hat{\alpha_i} \text{ and } i \in \mathcal{N}_{(\{o_t\})} \right\} \tag{11}$$

where $\mathcal{N}_{(\{o_t\})}$ stores all meta graph-based neighbors of items in o_t. To the end, our candidate set $\mathcal{A}_t$ is of high sample efficiency since it filters out irrelevant items (i.e., actions) while dynamically adapting to the user state shift.

4.3 Unbiased Off-policy Learning

The final stage of the off-policy learning is to utilize the state s_t and the candidate sets $\mathcal{A}_t$ to learn the target policy π_θ that maximizes the cumulative rewards $R(\pi_\theta)$ as in Eq. (1). For Top-K recommendation training, we implement a *two-stage policy learning* strategy [23] with a candidate generation model followed by ranking model to learn the target policy π_θ. Moreover, since the trained target policy π_θ holds a different distribution from the logging policy π_0 in the off-policy setting[6, 16], we further take advantages of *Counterfactual Risk Minimization* (CRM) [31] to correct the distribution discrepancy for achieving the unbiased off-policy optimization.

[2] In our experiment, we used a fixed initial state distribution, where $s_0 = 0 \in \mathbb{R}^d$

4.3.1 Top-K Recommendation Policy. In the training phrase, let $\mathcal{A}_t \in \mathcal{A}$ denote the current candidate set at time t, where $\mathcal{A}_t$ is derived from our candidate selection strategy from Eq. (11). The candidate generation model can be parameterized as a probability over all possible candidates conditioned on the current user state: $p_\theta(\mathcal{A}_t \mid s_t)$. The ranking model delivers the final recommendation results through optimizing together with the candidate generation model, which is drawn from a probability over all action a_t conditioned on the current state s_t and a candidate set $\mathcal{A}_t$, denoted by $q_\theta(a_t \mid s_t, \mathcal{A}_t)$. Assuming that the target policy takes a function form π_θ parameterized by $\theta \in \mathbb{R}^d$, the target policy can expressed as the following two-stage policy learning function:

$$\pi_\theta(a_t \mid s_t) = \sum_{\mathcal{A}_t} p_\theta(\mathcal{A}_t \mid s_t)\, q(a_t \mid s_t, \mathcal{A}_t) \tag{12}$$

With the formal definition of the target policy in Eq. (12), we can leverage policy gradient methods to derive the optimal target policy that maximizes expected long-term user satisfaction.

4.3.2 Policy Optimization. Remember that the state representations s_t are extracted from actions taken at each time t along the logging trajectory, i.e., π_0. However, the trained target policy π_θ holds a different distribution from the logging policy π_0, which has been proved in previous works [23, 26]. Directly optimizing target policy with conventional policy gradient methods would result in a biased recommender system [1]. In our work, we apply *Counterfactual Risk Minimization* (CRM) [31] to correct the discrepancy between the target policy π_θ and logging policy π_0, thus achieving unbiased off-policy learning. The *Counterfactual Risk Minimization* implements an Inverse Propensity Scoring (IPS) method to directly model the distribution shift between π_θ and π_0 in its objective. Formally, the counterfactual risk minimization objective is derived by applying the IPS in the expected cumulative rewards $R(\pi_\theta)$:

$$\widehat{R}(\pi_\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)\right] = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} \gamma^t \left\{\frac{\pi_\theta(a_t \mid s_t)}{\pi_0(a_t \mid s_t)}\right\}\right] \tag{13}$$

where $\frac{\pi_\theta(a_t|s_t)}{\pi_0(a_t|s_t)}$ is called the *importance weight*, which is used for balancing the empirical risk estimated from the π_0 as it came from the true risk of π_θ.

Feeding Eq. (12) into Eq. (13), the expected cumulative rewards of policy optimization can then be rewritten as:

$$\widehat{R}(\pi_\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} \gamma^t \left\{\frac{\sum_{\mathcal{A}_t} p_\theta(\mathcal{A}_t \mid s_t)\, q(a_t \mid s_t, \mathcal{A}_t)}{\pi_0(a_t \mid s_t)}\right\}\right] \tag{14}$$

Finally, the policy gradient of the cumulative reward function in Eq. (14) w.r.t. θ can be expressed as the following REINFORCE [36] gradient:

$$\nabla_\theta \widehat{R}(\pi_\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} \gamma^t \left\{\frac{\sum_{\mathcal{A}_t} q_\theta(a_t \mid s_t, \mathcal{A}_t)\, \nabla_{\theta p_\theta}(\mathcal{A}_t \mid s_t)}{\pi_0(a_t \mid s_t)}\right\}\right] \tag{15}$$

5 EXPERIMENTS

To thoroughly evaluate the proposed off-policy method for the recommendation, we conduct extensive experiments to answer the following research questions:

- **RQ1:** How does our MGpolicy perform compared with state-of-the-art off-policy recommendation methods?
- **RQ2:** Do different components (i.e., meta graph-based state representation, neighbor attentive candidate selection, counterfactual risk minimization for unbiased learning) help MGpolicy to achieve sparsity alleviation, sample efficiency, and bias alleviation.
- **RQ3:** How do hyper-parameters in MGpolicy impact the recommendation performance?

5.1 Experimental Setup

5.1.1 Online Environment Simulation. To train off-policy learning methods, we adopt two benchmark datasets from different recommendation domains, namely `MovieLens` [3] and `Douban book` [4]. For both datasets, we binarize the feedback data (i.e., ratings) by converting ratings of 4 or higher as the positive feedback (i.e., $r = 1$), otherwise negative (i.e., $r = 0$). The detailed statistics of both datasets are given in Table 1. Since these datasets can only reflect partial feedback (i.e., reward) information, it is infeasible to directly apply them to train off-policy learning recommenders due to the lack of ground-truth feedback. To facilitate the utility of the two binarized datasets in off-policy learning, we design the simulation environment based on an online simulator [2, 44, 46] to recover the missing reward r. The simulator takes the user-item pair (u_i, a_i) and their features as input and predicts the immediate feedback r_i. Our well-trained [5] simulator can then serve as a proxy of the real online environment, as it can give feedback for all the user-action pairs. As such, we can generate the recovered missing feedback in datasets by padding missing rewards with the trained simulator. The logged feedback samples are then acquired by running a logging policy π_0 on the recovered datasets. We adopt the wildly used uniform-based logging policy which samples each action at every interaction uniformly at random. It assumes every action's probability of being exposed is $\pi_{\text{uniform}}(a \mid s) = \frac{1}{|\mathcal{A}|}$.

Table 1: Statistics of the datasets. Density is computed by $\#Feedback/(\#Users \cdot \#Items)$.

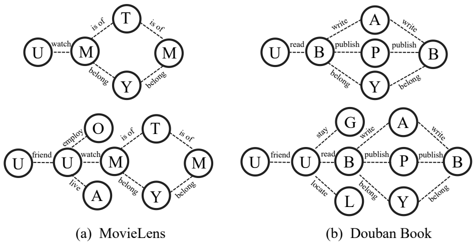
Dataset	#Users	#Items	#Total Feedback	#Feedback Per Customer	#Feedback Per Item	Density
`MovieLens`	6040	3883	1000209	165.5975	257.5866 %	4.26%
`Douban Book`	13024	22347	792062	60.8156	35.4438 %	0.27%

5.1.2 Meta graph information. In addition to the user-item interactions for constructing the logged feedback samples, we need to harness rich context information among meta graphs of the considered datasets, i.e., `MovieLens` and `Douban book`. These two datasets contain multiple heterogeneous relations for both users and items, i.e, user and item nodes connected with more than one type of heterogeneous neighbors, thus can provide rich context information for off-policy learning. We depict the meta graphs of `MovieLens` and `Douban book` in Figure 4. To ensure the meta graph quality, we only select meta graphs that allow the in- and out-degree of nodes (except the source and target node) larger than 1. Besides,

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

[4] http://book.douban.com

[5] Test results of overall 90% precision for immediate feedback prediction task.



(a) MovieLens (b) Douban Book

Figure 4: Meta graphs used in experiments.

we consider specific edge types between two nodes to capture the edge-type heterogeneity.

5.1.3 Baselines. We compare MGpolicy with 8 baselines in the recommendation task, which can be roughly sorted into (1) conventional HIN-enhanced methods, (2) off-policy learning methods and (3) context-enhanced RL methods. The category 1 leverages side information in the HINs to enhance the conventional recommendation. The category 2 uses off-policy learning to boost the recommendation while category 3 enhances RL-based recommendation with contexts from external resources (e.g., HINs and KGs).

- *HGT* [12](1): leverages meta relations to model heterogeneous information of HIN with Transformer architecture.
- *HAN* [34](1): leverages Graph Neural Network (GNN) to model context information from meta paths and implements hierarchical attentions on nodes and meta paths.
- *MCrec* [11](1): leverages Deep Neural Network (DNN) to model meta path-based context and propagates the context to user and item representations with co-attention mechanism.
- *Bandit-MLE* [10](2): is a value-based method that estimates rewards through Maximum Likelihood Estimation (MLE), then generates target policy by selecting actions with maximum rewards.
- *POEM* [31] (2): is a policy-based method that learns the target policy through maximizing the cumulative reward. It corrects the bias with an IPS estimator.
- *BanditNet* [15] (2): extends the policy-based off-policy learning to DNNs and optimizes a SNIPS estimator.
- *HUCB* [42] (3): is a HIN-enhanced bandit learning method. It uses HIN to assist the estimation of the upper confidence bound (UCB) of the cumulative reward.
- *KGQR* [45] (3): models KG information with Graph Convolution Network(GCN). It learns target policy with KG-enhanced state representations through the deep Q-network.

We evaluate all baselines using Mean Average Precision (MAP)@K and Normalized Discounted Cumulative Gain (NDCG)@K with $K = [5, 10, 20, 40]$. The Wilocoxon signed-rank test [37] is performed to evaluate whether the difference between MGpolicy and other baselines is significant.

5.1.4 Parameter Settings. In MGpolicy, the logged user feedback samples are generated through our online simulator with the uniform-based logging policy, while are split into train/test/validate set with a proportion of 60%/20%/20%. The same online simulation and sample split are also applied in all off-policy baselines. We set the layer number of the meta graph attention network in MGpolicy with $L = 3$ and the attention head number with $h = 8$. All neural networks-based (i.e., Transformer, GNN, DNN and GCN) baselines also keep 3 layers. We use $d = 128$ as the embedding dimension throughout baselines and our MGpolicy, while the candidate size $n = 20$ is set for MGpolicy and KGQR. For policy optimization, MGpolicy optimizes the two-stage policy gradient with AdaGrad [5], the same gradient descent method is also applied in all baselines. The hyper-parameters of all models are chosen by the grid search, including learning rate, batch size, L_2 norm regularization, discount factor γ, etc. The maximum epoch N_{epoch} for all methods is set as 2000, while an early stopping strategy is performed (i.e., if the loss stops to increase, then terminate the model training). We train all baselines and our MGpolicy on a Linux server with NVIDIA RTX 3090Ti GPU while testing the performance of the trained models with ranking length $K = [5, 10, 20, 40]$.

5.2 Performance Comparison (RQ1)

Table 2 reports the experimental results averaged over 5 repeated experiments. Overall, MGpolicy consistently yields the best performance among all datasets on both evaluation metrics. In addition, MGpolicy improves the strongest baseline for ranking metric NDCG@K by 5.9% and 43.7% [6] on MovieLens and Douban Book, respectively. For accuracy metric, MGpolicy improves MAP@K by 5.6% and 37.8% on the two datasets, respectively. This demonstrates that our MGpolicy indeed improves Top-K recommendation and is of superior accuracy and ranking capability.

Across the datasets, our MGpolicy achieves remarkable improvements on extreme sparse Douban Book (i.e., 0.27% density) compared with it on MovieLens (i.e., 4.26% density). While the performance of baselines deteriorates from MovieLens to Douban Book, MGpolicy continues to perform stably on Douban Book and improves NDCG@K and MAP@K by 43.7% and 37.8%, respectively. This is mainly because the baselines cannot handle the data sparsity, whereas our MGpolicy uses the context information in meta graphs to boost off-policy learning and can thus handle the sparsity well.

Among baselines, POEM outperforms all baselines on MovieLens in most cases, while KGQR is the strongest baseline for Douban Book. In addition, conventional HIN-enhanced methods cannot outperform other reinforcement learning methods. Although BanditNet uses an advanced SNIPS estimator compared with the IPS estimator leveraged in POEM, it still cannot outperform POEM. We infer this is because SNIPS introduces control variate to the IPS estimator that heavily penalizes the target policy, which limits the exploration ability of policy learning. Our MGpolicy employs the counterfactual risk minimization with an IPS-based estimator to efficiently reduce bias. KGQR alleviates the sparsity issue in Douban Book by modeling the side knowledge in KG. This indicates that the rich side information in external resources plays an important role in reinforcement learning to guide satisfying recommendations. However, KGQR cannot outperform MGpolicy since it ignores alleviating biases in the interactive recommendation setting. Our MGPolicy leverages rich context information to correct off-policy bias, thus can achieve satisfactory results in the recommendation.

[6] Calculated by averaging the improvement percentages under all K

Table 2: Performance comparison: bold numbers are the best results, best baselines are marked with underlines.

Datasets	Metrics	HGT^{rec}	HAN^{rec}	MCrec	Bandit-MLE	BanditNet	POEM	HUCB	KGQR	MGpolicy	Improv.%
MovieLens	MAP@5	0.444	0.421	0.405	0.721	0.841	0.825	0.611	0.721	**0.944***	12.2%
	MAP@10	0.533	0.512	0.509	0.761	0.852	0.891	0.635	0.778	**0.939***	5.4%
	MAP@20	0.613	0.579	0.577	0.751	0.882	0.908	0.662	0.824	**0.917***	1.0%
	MAP@40	0.725	0.701	0.678	0.764	0.838	0.892	0.697	0.855	**0.927**	3.9%
	NDCG@5	0.572	0.531	0.512	0.822	0.748	0.821	0.632	0.736	**0.931***	13.3%
	NDCG@10	0.657	0.589	0.556	0.871	0.822	0.828	0.666	0.805	**0.926***	6.3%
	NDCG@20	0.708	0.656	0.633	0.908	0.851	0.915	0.691	0.849	**0.926***	1.2%
	NDCG@40	0.756	0.721	0.705	0.901	0.829	0.903	0.711	0.866	**0.928***	2.8%
Douban Book	MAP@5	0.397	0.376	0.365	0.642	0.623	0.591	0.519	0.648	**0.931***	43.7%
	MAP@10	0.451	0.424	0.419	0.625	0.608	0.571	0.542	0.571	**0.907***	45.1%
	MAP@20	0.522	0.478	0.469	0.596	0.571	0.538	0.581	0.629	**0.889***	41.3%
	MAP@40	0.614	0.599	0.572	0.604	0.612	0.546	0.656	0.723	**0.875***	21.0%
	NDCG@5	0.614	0.599	0.572	0.588	0.567	0.612	0.521	0.571	**0.944***	53.7%
	NDCG@10	0.552	0.446	0.432	0.553	0.548	0.594	0.533	0.628	**0.929***	47.9%
	NDCG@20	0.617	0.555	0.529	0.549	0.528	0.572	0.583	0.650	**0.907***	39.5%
	NDCG@40	0.653	0.602	0.581	0.538	0.572	0.608	0.653	0.693	**0.927***	33.8%

$*$ indicates statistically significant improvements (measured by Wilocoxon signed-rank test at $p < 0.05$) over all baselines.

5.3 Analysis of MGpolicy (RQ2)

We conduct an in-depth analysis of the effectiveness of our MGpolicy on mitigating sparsity issue, sample inefficiency, and bias issue. Our MGpolicy includes three important components, namely, meta graph-based state representation (cf. Section 4.1.2), neighbor attentive candidate selection (cf. Section 4.2), counterfactual risk minimization for unbiased learning (cf. Section 4.3). We evaluate the performance of our MGpolicy with different variant combinations and show our observations in this section.

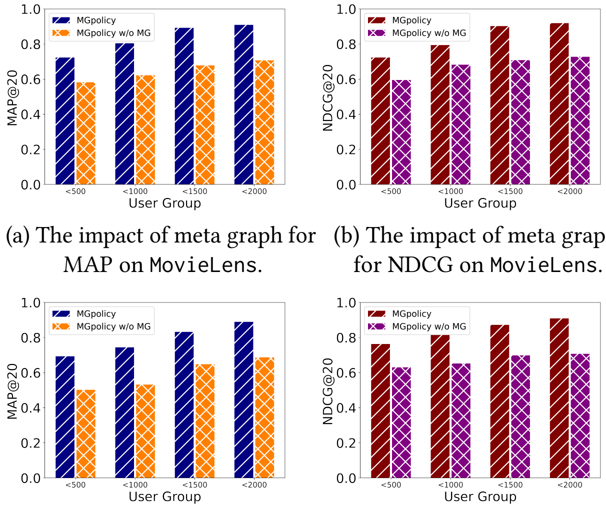
5.3.1 Sparsity alleviation with meta graphs. We investigate whether exploiting rich context information in meta graphs benefits for achieving better recommendation from sparse logged data. To show how meta graph information works in MGpolicy, we train the MGpolicy with or without meta graph-based state representation component, namely, *MGpolicy* or *MGpolicy-w/o MG*. The item representations in the *MGpolicy* are trained from meta graphs with our meta graph attention network, whereas *MGpolicy-w/o MG* leverages ID-based item representations acquired from the Matrix Factorization [17]. Table 3 shows the performance of the *MGpolicy* and *MGpolicy-w/o MG* on MovieLens and Douban Book. To represent different sparsity levels, we divide users in the test set into four groups based on the interaction number per user. For example, user group 1 represents the sparsest data in which users have less than 500 ratings for movies; likewise user group 2, 3 and 4 have less than 1000, 1500 and 2000 ratings, respectively. With the trained *MGpolicy* and *MGpolicy-w/o MG* model, we give the test results of the two models on the four user groups. The results are given in Figure 5 and we have the following observations.

As shown in Table 3, the eliminating of meta graph information in *MGpolicy-w/o MG* model results in a deteriorated performance compared with *MGpolicy* equipped with meta graph-based state representation component. This indicates that the meta graph-based state representation component has a significant contribution to off-policy learning. Figure 5 shows that the recommendation performance of *MGpolicy* consistently outperforms *MGpolicy-w/o MG*. Moreover, the meta graph-based state representation improves recommendation more significantly on the sparser dataset Douban Book. In addition, *MGpolicy* improves *MGpolicy-w/o MG* by the largest margin on the sparsest user group 1. These promising findings suggest the superiority of applying meta graph-based state representation in off-policy learning to achieve satisfying recommendations, especially in the data sparsity scenario.

Table 3: Ablation Study of MGpolicy. The number after ± indicates the improvement/deterioration percentage of the variant compared with *MGpolicy*.

Dataset	Metrics	MGpolicy	MGpolicy-w/o MG	MGpolicy-w/o ATT	MGpolicy-CE
MovieLens	MAP@5	0.944	0.732 (-29.0%)	0.846 (-11.6%)	0.880 (-7.3%)
	MAP@10	0.939	0.711 (-32.1%)	0.873 (-7.6%)	0.883 (-6.3%)
	MAP@20	0.917	0.708 (-29.5%)	0.853 (-7.5%)	0.875 (-4.8%)
	MAP@40	0.927	0.701 (-32.2%)	0.804 (-15.3%)	0.871 (-6.4%)
	NDCG@5	0.931	0.746 (-24.8%)	0.839 (-11.0%)	0.874 (6.5%)
	NDCG@10	0.926	0.739 (-25.3%)	0.825 (-12.2%)	0.893 (-3.7%)
	NDCG@20	0.926	0.731 (-26.7%)	0.885 (-4.6%)	0.907 (-2.1%)
	NDCG@40	0.928	0.727 (-27.6%)	0.868 (-6.9%)	0.901 (-3.0%)
Douban Book	MAP@5	0.931	0.709 (-31.3%)	0.828 (-12.4%)	0.849 (-9.6%)
	MAP@10	0.907	0.692 (-31.1%)	0.809 (-12.1%)	0.857 (-5.8%)
	MAP@20	0.889	0.683 (-30.2%)	0.816 (-8.9%)	0.842 (-5.6%)
	MAP@40	0.875	0.701 (-24.8%)	0.811 (-7.9%)	0.851 (-2.8%)
	NDCG@5	0.944	0.715 (-32.0%)	0.864 (-9.3%)	0.873 (-8.1%)
	NDCG@10	0.929	0.701 (-32.5%)	0.859 (-8.1%)	0.870 (-6.8%)
	NDCG@20	0.907	0.704 (-28.8%)	0.821 (-10.5%)	0.868 (-4.5%)
	NDCG@40	0.927	0.738 (-25.6%)	0.843 (-10.0%)	0.878 (-5.6%)

5.3.2 Sample efficiency of neighbor attentive selection . The sample efficiency in recommendation denotes the ability to reduce the huge item space by filtering out irrelevant items for efficient calculations, so as to improve the recommendation effectiveness [45]. Our developed neighbor attentive selection leverages attentive mechanism and structural information of meta graphs to promote an efficient action search, such that the sample efficiency can be achieved. To demonstrate the effectiveness of the attentive mechanism in neighbor attentive candidate selection on improving the sample efficiency of MGpolicy, we test MGpolicy without (w/o) the attentive mechanism (i.e., *MGpolicy-w/o ATT*), in which the probability of selecting one item into the candidate set is fixed as $\alpha_i = \frac{1}{n}$. We report the performance of *MGpolicy-w/o ATT* in Table 3. We observed that removing the attention mechanism leads to a downgraded performance of *MGpolicy* in Table 3, which validates the effectiveness of the attention mechanism in improving the recommendation. This is because the attention mechanism can filter out irrelevant



(a) The impact of meta graph for MAP on MovieLens.

(b) The impact of meta graph for NDCG on MovieLens.

(c) The impact of meta graph for MAP on Douban Book.

(d) The impact of meta graph for NDCG on Douban Book.

Figure 5: Effectiveness analysis on meta graph: different user groups control the interaction numbers.

items based on their contributions to the current state, resulting in improved sampling efficiency to enhance the recommendation.

5.3.3 Bias alleviation with counterfactual risk minimization. The Counterfactual Risk Minimization (CRM) loss (cf. Eq. (14)) in MGpolicy is designed for alleviating the bias caused by the distribution mismatch between logging policy and the learned target policy. To investigate the effectiveness of the CRM loss, we apply our MGpolicy with conventional cross-entropy (CE) loss (i.e., *MGpolicy-CE*) to show how it performs compared with MGpolicy with the CRM loss. We report the performance of *MGpolicy-CE* in Table 3 and compare the received cumulative rewards while training *MGpolicy* and *MGpolicy-CE* in Figure 6. Here are our observations: First, *MGpolicy* with CRM loss consistently achieves larger cumulative rewards than *MGpolicy-CE* on both datasets. Second, the cumulative rewards gained by *MGpolicy-CE* suffer from drastic fluctuations, which somehow indicates the unstable performance of *MGpolicy-CE*. The sub-optimal performance of *MGpolicy-CE* indicates that the bias issue in the off-policy learning can lead to a recommendation agent with downgraded and unstable performance. On the contrary, our MGpolicy takes advantage of counterfactual risk minimization to learn a high-quality target policy in a stable manner. Third, more iterations are needed for *MGpolicy-CE* to achieve stable cumulative rewards compared with *MGpolicy*. This indicates that *MGpolicy* can quickly reach stable states using a small number of iterations.

5.4 Parameter Sensitivity Analysis (RQ3)

In this section, we investigate how our performance is sensitive to the dimension (i.e., d) of state representation and the size (i.e., n) of candidate selection.

5.4.1 Sensitivity of state dimension. Figure 7 (a) (b) report the parameter sensitivity of our method w.r.t. state dimension d with $d = \{32, 64, 128, 256, 512, 1024\}$. Apparently, the performance of MGpolicy on MovieLens and Douban Book demonstrates increasing trends from $d = 32$, then reaches the peak when $d = 128$ and $d = 256$ respectively. This is reasonable since d controls the number of latent vectors of state representations, and low-dimensional latent vectors cannot retain enough information to assist in learning high-quality target policy. The performance of MGpolicy degrades slightly after the peak, and then becomes stable. This demonstrates that MGpolicy is robust under varying state dimensions.

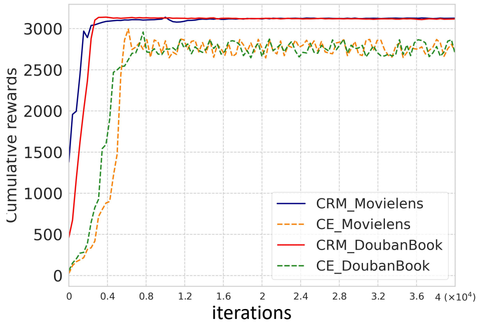
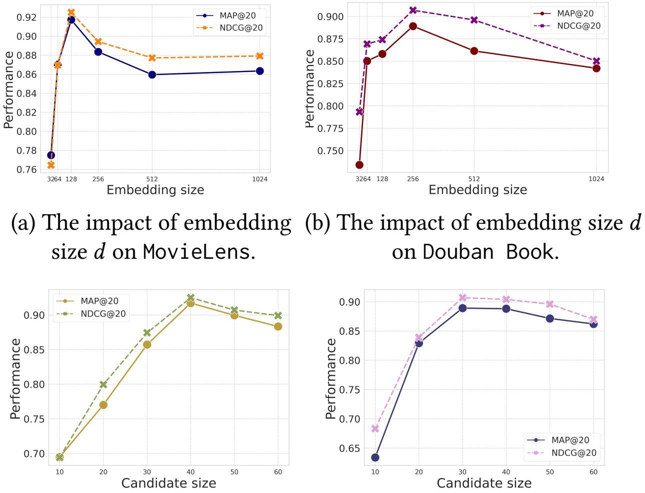


Figure 6: Learning curves of *MGpolicy-CE* and *MGpolicy*.

5.4.2 Sensitivity of candidate size. We study the sensitivity of our performance to the candidate size n in neighbor attentive candidate selection component. By varying n from $\{10, 20, 30, 40, 50, 60\}$ in Figure 7 (c) (d), we observe that the performance of MGpolicy first improves as candidate size increases on both MovieLens and Douban Book. The reason for the degraded performance with $n = 10$ is that the small size of the candidate pool would limit the exploration ability of finding the appropriate candidates for recommendation. Thereafter, we can witness the improvement of the performance by further increasing the candidate size. Although increasing candidate sizes can make classic Top-k recommendation models harder to give correct results, our neighbor attentive candidate selection can filter out irrelevant items to make our MGpolicy perform effectively for the large action space. Then, the performance of MGpolicy decreases after the peak points of $n = 40$ and $n = 30$ on MovieLens and Douban Book. This is reasonable since more negative instances would exist in a larger candidate pool, which offers more chances for the agent to recommend negative items to users.

6 CONCLUSION AND FUTURE WORK

In this work, we proposed MGpolicy to tackle a novel challenge of off-policy bias correction by leveraging contextual information in meta graphs. We designed a meta graph attention network to learn rich contextual information through modeling complex semantics and heterogeneity of meta graphs. By enriching state representations with contextual information, the learned target policy is capable of inferring potential user feedback on actions. In addition, we reduce the huge candidate space by utilizing attention mechanism and structure information of meta graphs. Finally, MGpolicy corrects the off-policy bias with counterfactual risk minimization. Extensive experiments and analyses on two datasets demonstrate MGpolicys' abilities on mitigating sparsity issue, sample inefficiency and bias issue and yield improved recommendation performance. In future work, we plan to investigate MGpolicy with



(a) The impact of embedding size d on MovieLens.

(b) The impact of embedding size d on Douban Book.

(c) The impact of candidate size n on MovieLens.

(d) The impact of candidate size n on Douban Book.

Figure 7: Parameter sensitiveness: embedding size d controls the latent factor numbers of state representations; candidate size n controls the length of candidate set.

other off-policy optimization methods, such as the Actor-Critic that combines Q-learning and policy gradient.

REFERENCES

[1] Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H Chi. 2019. Top-k off-policy correction for a REINFORCE recommender system. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 456–464.

[2] Xiaocong Chen, Lina Yao, Julian McAuley, Guangling Zhou, and Xianzhi Wang. 2021. A Survey of Deep Reinforcement Learning in Recommender Systems: A Systematic Review and Future Directions. *arXiv preprint arXiv:2109.03540* (2021).

[3] Yifan Chen, Yang Wang, Xiang Zhao, Jie Zou, and Maarten De Rijke. 2020. Block-Aware Item Similarity Models for Top-N Recommendation. *ACM Transactions on Information Systems (TOIS)* 38, 4 (2020), 1–26.

[4] Aminu Da'u and Naomie Salim. 2020. Recommendation system based on deep learning methods: a systematic review and new directions. *Artificial Intelligence Review* 53, 4 (2020), 2709–2748.

[5] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* 12, 7 (2011).

[6] Louis Faury, Ugo Tanielian, Elvis Dohmatob, Elena Smirnova, and Flavian Vasile. 2020. Distributionally robust counterfactual risk minimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 3850–3857.

[7] Seth Flaxman, Sharad Goel, and Justin M Rao. 2016. Filter bubbles, echo chambers, and online news consumption. *Public opinion quarterly* 80, S1 (2016), 298–320.

[8] C Lee Giles, Gary M Kuhn, and Ronald J Williams. 1994. Dynamic recurrent neural networks: Theory and applications. *IEEE Transactions on Neural Networks* 5, 2 (1994), 153–156.

[9] Alexandre Gilotte, Clément Calauzènes, Thomas Nedelec, Alexandre Abraham, and Simon Dollé. 2018. Offline a/b testing for recommender systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 198–206.

[10] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. 2013. *Applied logistic regression*. Vol. 398. John Wiley & Sons.

[11] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1531–1540.

[12] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In *Proceedings of The Web Conference 2020*. 2704–2710.

[13] Zhipeng Huang, Yudian Zheng, Reynold Cheng, Yizhou Sun, Nikos Mamoulis, and Xiang Li. 2016. Meta Structure: Computing Relevance in Large Heterogeneous Information Networks. In *KDD*. 1595–1604. https://doi.org/10.1145/2939672.2939815

[14] Olivier Jeunen and Bart Goethals. 2021. Pessimistic reward models for off-policy learning in recommendation. In *Fifteenth ACM Conference on Recommender Systems*. 63–74.

[15] Thorsten Joachims, Adith Swaminathan, and Maarten de Rijke. 2018. Deep learning with logged bandit feedback. In *International Conference on Learning Representations*.

[16] Yonghan Jung, Jin Tian, and Elias Bareinboim. 2020. Learning causal effects via weighted empirical risk minimization. *Advances in neural information processing systems* 33 (2020).

[17] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[18] Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. 2019. Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction. arXiv:1906.00949 [cs.LG]

[19] Pushpendra Kumar and Ramjeevan Singh Thakur. 2018. Recommendation system techniques and related issues: a survey. *International Journal of Information Technology* 10, 4 (2018), 495–501.

[20] Qian Li, Wenjia Niu, Gang Li, Yanan Cao, Jianlong Tan, and Li Guo. 2015. Lingo: linearized grassmannian optimization for nuclear norm minimization. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 801–809.

[21] Qian Li, Xiangmeng Wang, and Guandong Xu. 2021. Be Causal: De-biasing Social Network Confounding in Recommendation. *arXiv preprint arXiv:2105.07775* (2021).

[22] Qian Li and Zhichao Wang. 2017. Riemannian submanifold tracking on low-rank algebraic variety. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[23] Jiaqi Ma, Zhe Zhao, Xinyang Yi, Ji Yang, Minmin Chen, Jiaxi Tang, Lichan Hong, and Ed H Chi. 2020. Off-policy learning in two-stage recommender systems. In *Proceedings of The Web Conference 2020*. 463–473.

[24] Yusuke Narita, Shota Yasui, and Kohei Yata. 2021. Debiased Off-Policy Evaluation for Recommendation Systems. In *Fifteenth ACM Conference on Recommender Systems*. 372–379.

[25] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. 2018. Image transformer. In *International Conference on Machine Learning*. PMLR, 4055–4064.

[26] Yuta Saito and Thorsten Joachims. 2021. Counterfactual Learning and Evaluation for Recommender Systems: Foundations, Implementations, and Recent Advances. In *Fifteenth ACM Conference on Recommender Systems*. 828–830.

[27] Aravind Sankar, Xinyang Zhang, and Kevin Chen-Chuan Chang. 2019. Meta-gnn: metagraph neural network for semi-supervised learning in attributed heterogeneous information networks. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. 137–144.

[28] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. 2016. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering* 29, 1 (2016), 17–37.

[29] James E. Smith and Robert L. Winkler. 2006. The Optimizers Curse: Skepticism and Postdecision Surprise in Decision Analysis. *Manage. Sci.* 52, 3 (mar 2006), 311–322. https://doi.org/10.1287/mnsc.1050.0451

[30] Adith Swaminathan and Thorsten Joachims. 2015. Batch learning from logged bandit feedback through counterfactual risk minimization. *The Journal of Machine Learning Research* 16, 1 (2015), 1731–1755.

[31] Adith Swaminathan and Thorsten Joachims. 2015. Counterfactual risk minimization: Learning from logged bandit feedback. In *International Conference on Machine Learning*. PMLR, 814–823.

[32] Flavian Vasile, David Rohde, Olivier Jeunen, and Amine Benhalloum. 2020. A Gentle Introduction to Recommendation as Counterfactual Policy Learning. In *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization*. 392–393.

[33] Chengwei Wang, Tengfei Zhou, Chen Chen, Tianlei Hu, and Gang Chen. 2020. Off-Policy Recommendation System Without Exploration. *Advances in Knowledge Discovery and Data Mining* 12084 (2020), 16.

[34] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *The World Wide Web Conference*. 2022–2032.

[35] Xiangmeng Wang, Qian Li, Wu Zhang, Guandong Xu, Shaowu Liu, and Wenhao Zhu. 2020. Joint relational dependency learning for sequential recommendation. *Advances in Knowledge Discovery and Data Mining* 12084 (2020), 168.

[36] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3 (1992), 229–256.

[37] R. F. Woolson. 2008. *Wilcoxon Signed-Rank Test.* John Wiley Sons, Ltd, 1–3. https://doi.org/10.1002/9780471462422.eoct979 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780471462422.eoct979

[38] Teng Xiao and Donglin Wang. 2021. A general offline reinforcement learning framework for interactive recommendation. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*.

[39] Fenfang Xie, Angyu Zheng, Liang Chen, and Zibin Zheng. 2021. Attentive Meta-graph Embedding for item Recommendation in heterogeneous information

networks. *Knowledge-Based Systems* 211 (2021), 106524.

[40] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*. PMLR, 5453–5462.

[41] Suiyun Zhang, Zhizhong Han, Yu-Kun Lai, Matthias Zwicker, and Hui Zhang. 2019. Stylistic scene enhancement GAN: mixed stylistic enhancement generation for 3D indoor scenes. *The Visual Computer* 35, 6 (2019), 1157–1169.

[42] Xiaoying Zhang, Hong Xie, and John CS Lui. 2021. Heterogeneous Information Assisted Bandit Learning: Theory and Application. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2135–2140.

[43] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. 2017. Meta-graph based recommendation fusion over heterogeneous information networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 635–644.

[44] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. Recommendations with negative feedback via pairwise deep reinforcement learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1040–1048.

[45] Sijin Zhou, Xinyi Dai, Haokun Chen, Weinan Zhang, Kan Ren, Ruiming Tang, Xiuqiang He, and Yong Yu. 2020. Interactive recommender system via knowledge graph-enhanced reinforcement learning. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 179–188.

[46] Lixin Zou, Long Xia, Pan Du, Zhuo Zhang, Ting Bai, Weidong Liu, Jian-Yun Nie, and Dawei Yin. 2020. Pseudo Dyna-Q: A reinforcement learning framework for interactive recommendation. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 816–824.