Off-policy Learning over Heterogeneous Information for Recommendation

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

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

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

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

ABSTRACT

Reinforcement learning has recently become an active topic in recommender system research, where the logged data that records interactions between items and users feedback is used to discover the policy. Much off-policy learning, referring to the procedure of policy optimization with access only to logged feedback data, has been a popular research topic in reinforcement learning. However, the log entries are biased in that the logs over-represent actions favored by the recommender system, as the user feedback contains only partial information limited to the particular items exposed to the user. As a result, the policy learned from such off-line logged data tends to be biased from the true behaviour policy.

In this paper, we are the first to propose a novel off-policy learning augmented by meta-paths for the recommendation. We argue that the Heterogeneous information network (HIN), which provides rich contextual information of items and user aspects, could scale the logged data contribution for unbiased target policy learning. Towards this end, we develop a new HIN augmented target policy model (HINpolicy), which explicitly leverages contextual information to scale the generated reward for target policy. In addition, being equipped with the HINpolicy model, our solution adaptively receives HIN-augmented corrections for counterfactual risk minimization, and ultimately yields an effective policy to maximize the long run rewards for the recommendation. Finally, we extensively evaluate our method through a series of simulations and large-scale real-world datasets, obtaining favorable results compared with state-of-the-art methods.

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

CCS CONCEPTS

• **Information systems**; • **Collaborative filtering**; • **Computer systems organization** → Robotics; • **Networks** → Network;

KEYWORDS

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

ACM Reference Format:
Xiangmeng Wang, Qian Li, Dianer Yu, and Guandong Xu. 2018. Off-policy Learning over Heterogeneous Information for Recommendation. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Recommender system (RS) has become a panacea to assist users in discovering preferred contents from the massive information provided[15, 33]. Traditional approaches to recommendation are often based on form of collaborative filtering [35, 46] or knowledge-based systems [27]. Since traditional RS methods are static, they can not well handle the sequential and dynamic nature of user interaction with the system [1], which is the essence of RS. Recently, reinforcement learning (RL)-based approaches have attracted a lot of attention in recommender systems, in which an agent (recommender) is guided by a recommendation algorithm (policy) to drive user interaction with the environment [34, 37, 57, 58]. The core idea of RL methods is to train RS as an intelligent agent that learns an optimal recommendation policy to maximize each user's long-term satisfaction with its system [9]. To train such an optimal RL agent, it is natural to perform online learning on interactions between recommender systems and users. However, such online learning is infeasible in real RS since it might degrade user satisfaction and deteriorate the revenue of the platform [6, 23, 51]. Fortunately, off-policy learning emerges as a favorable opportunity for policy optimization, which uses historical user feedback instead of constructing expensive online interactive environments [29, 41, 57].

Off-policy learning has attracted increasing interest in recommender system research in recent years [43, 52, 53]. As shown in Figure 1, the goal of off-policy learning is to maximize each user's long term satisfaction with the system, given logged data generated by the logging policy of the recommender system. Achieving this

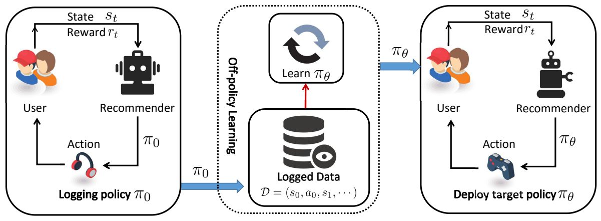


Figure 1: Off-policy learning in Recommendation.

off-policy goal has to address the question of how much reward would be received if a new target policy had been deployed instead of the original logging policy. This counterfactual question is not easy to address, since the target policy is different from the historical logging policy in the off-policy setting [48, 49]. To this effect, most off-policy learning for recommendation relies on inverse propensity score (IPS) estimator correction to get an unbiased empirical risk minimization objective [10, 39]. A major disadvantage of these methods is that IPS is likely to be over-fitted as some actions have zero probability of being taken in recommender systems [19, 30, 47]. For example, 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 ubiquity of biases [5] (e.g., exposure bias or conformity bias). Thus a large number of actions would not be selected at all in recommender systems, leading to the "poor gets poorer" phenomenon [21], i,e., a video will not be nominated in the target policy simply because it was never nominated in the behavior logging policy.

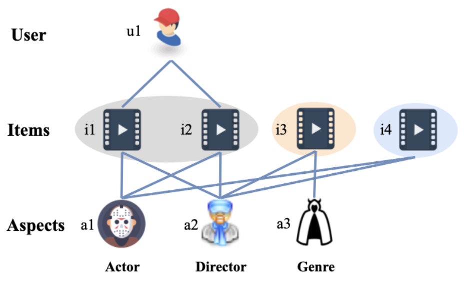


Figure 2: A toy example of inferring rewards from HIN.

Towards this end, we claim that real-world context information could be useful to deal with deficient log data and empower off-policy learning in the recommendation. An example is shown in Figure 2, we have a user u_1 exposed to (i_1, i_2) but not to (i_3, i_4). As a result, it is impossible to learn cumulative rewards (e.g., users' feedback during a period of time) for a policy which selects actions (e.g., i_3 and i_4) not contained in the logged data. That means information about the reward for the action of interacting with i_3 and i_4 can never be chosen by the deterministic logging policy. Fortunately, we can infer such feedback information with the assistance of contextual information. Suppose u_1 offers positive feedback to i_1 and i_2. Since both i_1 and i_2 have the same actor a_1 and the same director a_2, we may infer that the combination of actor a_1 and director a_2 is an important factor of u_1's interest. The movie i_4 with the same actor a_1 and director a_2 should be highly likely to be preferred by u_1. By contrast, u_1 probably has less interested on i_3 with a different actor. That means i_3 and i_4 could offer high-quality negative feedback and positive feedback of i_4. As such, exploiting the contextual information can alleviate the "poor gets poorer" phenomenon in off-policy learning for the recommendation.

In this paper, we investigate how to correct off-policy biases with the assistance from Heterogeneous Information Network, i.e.,*HIN augmented off-policy learning (HINpolicy)*. In particular, we design a co-attentive mechanism to mutually derive the interaction-specific context information to produce the high-quality target policy of the recommendation. Meanwhile, the counterfactual risk minimization is designed to explore the target policy, so as to optimize the behaviour policy that can maximize users' long-term satisfaction. To summarise, our method offers the following contributions [1]:

- We are the first to leverage contextual information in HIN to provide high-quality target policy learning for correcting the bias in off-policy recommendations.
- We develop a new end-to-end framework HINpolicy, which achieves counterfactual risk minimization in an explicit manner under the co-attention mechanism.
- Empirically, we generate an online environment using simulators to carry out experiments on two benchmark datasets. Extensive results show that our methods outperform the state-of-the-art methods.

2 RELATED WORK

2.1 Traditional Recommendation

Being supervised by the historical records is the common practice in majority models, including traditional collaborative filtering [35], content-based filtering [46] and knowledge-based systems [27]. Another topic closely related to above categories is deep neural models, such as multi-layer perceptron [11], denoising auto-encoders [56], convolutional neural network (CNN) [2, 42, 50, 60], recurrent neural network (RNN) [28, 54], memory network [8, 25] and attention architectures [7]. Based on the partially observed history dataset, these existing models usually predict a customer's feedback by a learning function to maximize some well-defined evaluation metrics in ranking, such as Recall, Precision and NDCG. However, most of them are myopic because the learned policies are greedy with estimating customers' feedback and unable to optimize customers' feedback in the long run.

2.2 Off-policy Learning for Recommendation

Reinforcement algorithms can be generally divided into on-policy and off-policy methods [9]. Off-policy recommendation learning from logged user feedback has attracted increased attention in recent years [6, 31, 39], which falls into two categories. The first approach to off-policy uses inverse propensity scoring (IPS) to correct the selection bias caused by the discrepancy between the target policy and the logging policy. Swaminathan et al. [49] propose an Optimizer for Exponential Models (POEM) that learns target policy free from propensity overfitting based on IPS estimator. Joachims

[1]Our code is currently available at https://www.dropbox.com/sh/s6bsinvu6z8n000/AAAV66XJZkZLHQVVZRsE6Apca?dl=0

et al. [31] propose a BanditNet that includes an additional self-normalisation IPS (SNIPS) term while they extend the off-policy learning to deep neural networks, which is a recent notable extension. Ma et al. [39] formulate the target policy of the two-stage recommender system as the composition of candidate generation policy and the ranker policy, meanwhile deriving the importance weight based on the IPS to correct the candidate generation policy. An alternative is the value-based approaches for the off-policy recommendation, such as Q-learning algorithms. They in principle aim to learn a state-action function, and then use this function to directly recover the optimal policy. Xin et al. adopt a Q-learning perspective to deal with sequential recommendation tasks, exploiting both self-supervised (organic) and reinforcement (logged data) signals [59]. Analogously, Sakhi et al. proposed a probabilistic latent model that combines organic and logged data in a Bayesian value-based manner [44].

The problems of off-policy learning are generally pervasive and challenging in RL, and in recommender systems in particular. In the scenario of recommender systems, item catalogues and user behaviour change rapidly, substantial policy changes are required. Therefore it is not sufficient to take the classic user interaction history to constrain the policy updates. This work tackles the problem of off-policy learning augmented by Heterogeneous information, which has not been well studied.

3 PRELIMINARY

3.1 Off-policy Learning for Recommendation

Unlike classical reinforcement learning, off-policy learning does not have real-time interactions with recommender systems due to learning and infrastructure constraints. Instead, in off-policy learning setting for the recommendation, we have access to a logged dataset of trajectories $\mathcal{D}$. The generation of $\mathcal{D}$ can be formulated with a Markov Decision Process (MDP) as denoted in Figure 1, 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 to the user state s;
- γ: a discount factor $\gamma \in [0,1]$ used for future immediate rewards;

Particularly, $\mathcal{D} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma\}$ has been collected under stochastic logging policy $\pi_0(a|s)$ that describes a probability distribution over items $\mathcal{A}$ (i.e., action), conditioned on user states $\mathcal{S}$. Meanwhile, the recommender system receives feedback reward $r(a,s)$ (i.e., clicks or watch time) for this particular state-action pair. As a result, training a recommender system seeks a policy π_θ that maximizes the expected cumulative rewards $R(\pi_\theta)$ over potentially infinite time horizon T, is defined as

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

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 Heterogeneous Information Network

Apart from the logged trajectories, we have additional contextual information about users and actions, e.g., social relations between users or actions (items)' genres. We aim to incorporate such contextual information in the Heterogeneous information network (HIN) into each state-action pair to optimize the recommendation policy. Heterogeneous information network (HIN), which records the complex relations between multiple types of involved entities, can be used to introduce useful information into state s_t to guide a better learning along policy $\pi_\theta(a_t|s_t)$. The formal definitions of HIN and Meta-path are given in Appendix A.1. In our setting, the HIN information is incorporated into the state vector s_t to learn the influence between user's context information and preference shift. Previous off-policy learning has frequently assumed that the choosing of presented actions depends only on user's descriptions (e.g., user id) and its historical interactions with items, leaving the influence of the large volume of unobserved user and item attributes behind.

4 METHODOLOGY

The overall framework of HINpolicy is presented in Figure 3, which consists of two important components, the HIN-augmented policy learning and counterfactual risk minimization. In HIN-augmented policy learning, we aim at leveraging complex relations in meta-paths to learn the meta-path context for the involved users and actions, then use the designed co-attention mechanism to derive the context-aware state that guides a better policy learning for the recommendation. We further take advantage of counterfactual risk minimization for the unbiased approximation of policy evaluation. Bias is ubiquitous in the off-policy learning setting [23], since the logged feedback data generated by a historical behaviour policy π_0 of the recommender system is different from the target policy π_θ trained. To correct the bias of distribution mismatch between behaviour policy and target policy, counterfactual risk minimization (CRM) [49] uses Inverse Propensity Scoring (IPS) estimator to re-weight the logged data according to ratios of slate probabilities under the target and logging policy. The final corrected recommendation policy π_θ is then used to produce high-quality candidate actions that wait to be re-ranked by the Top-K ranking model.

4.1 HIN-augmented Policy Learning

While off-policy learning methods [39, 63] achieve great success towards policy optimization in recommendations, the benefits of contextual information are not fully explored to improve policy learning. Heterogeneous information network (HIN), whose nodes are of different types and links among nodes represent different relations, reveals high-order dependency in recommendation environments (e.g., users' behaviours, recommendation policies, and action' aspects). Towards this, we take HIN as the prior knowledge of the policy learning to exploit its rich relations for exploring more suitable positive feedback that is missing in the logged data. In this section, we propose HIN-augmented policy learning that learns

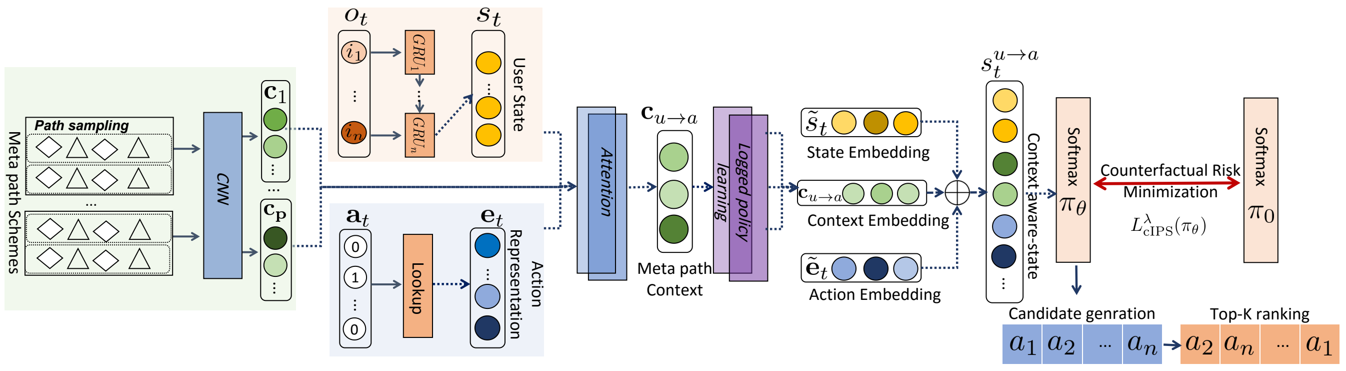


Figure 3: Our model framework of HINpolicy. Our HINpolicy includes rich contextual information retained in meta-path schemes for policy learning, thus capturing the potential influence from user/item attributes.

context-aware state by updating user state representation, meta-path context and action representation in a mutual enhancement way, so as to guide a better target policy learning.

4.1.1 State Representation. To derive the impact of meta-path based context towards the preference shift of users on taking actions, we first categorize the user state s_t with key information about the user preference. In this subsection, we design the state representation module, which extracts user's preferences through its historical interactions with items. Generally, the state representation s_t at time t in online recommendation scenario is learned from the user's interactions (e,g., clicked) at timestep t.

Formally, for state s_t, we have a set of actions $o_t = \{i_1, i_2, ..., i_n\}$ interacted by the user. Considering $\{o_t\}$ have sequential patterns, we resort Recurrent Neural Networks (RNN) [22] to learn an embedding vector $\mathbf{o}_t \in \mathbb{R}^d$ from $\{o_t\}$. To aggregate user's historical embedding $\mathbf{o}_t$, we conducted experiments with a large volume of popular RNN cells, including Bidirectional Recurrent Neural Networks (BRNN) [45], Gated Recurrent Units (GRU) [12], and Long Short-Term Memory family (LSTM) [24] with varying gates. Finally, the RNN with a gated recurrent unit (GRU) stands out among these cells due to its stability and computational efficiency. Hence, we learn the representation of the user state s_t by a GRU cell:

$$\begin{aligned}
\mathbf{z}_t &= \sigma_g\left(\mathbf{W}_1\mathbf{o}_t + \mathbf{U}_1 s_{t-1} + \mathbf{b}_1\right)\\
\mathbf{r}_t &= \sigma_g\left(\mathbf{W}_2\mathbf{o}_t + \mathbf{U}_2 s_{t-1} + \mathbf{b}_2\right)\\
\hat{s}_t &= \sigma_h\left(\mathbf{W}_3\mathbf{o}_t + \mathbf{U}_3\left(\mathbf{r}_t \circ s_{t-1}\right) + \mathbf{b}_3\right)\\
s_t &= (1-\mathbf{z}_t)\circ s_{t-1} + \mathbf{z}_t \circ \hat{s}_t
\end{aligned} \tag{2}$$

where $\mathbf{z}_t$ and $\mathbf{r}_t$ denote the update gate and reset gate vector generated by GRU, $\circ$ 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 representation of the current user state.

4.1.2 Attentive Meta-path Context Representation. The attentive meta-path context representation module produces interaction-specific context that captures diverse semantics of meta-paths on user-action interactions. Our attentive meta-path context representation module calculates attention weights over meta-paths conditioned on state-action pairs, thus can capture the influence of each meta-path on user interest drift.

In attentive meta-path context representation module, we first resort to *Meta-path Based Random Walks* [16] for generating path instances $\rho = \{v_1, \cdots, v_l\}$ under a specific meta-path scheme $\mathbf{p}$ through an effective meta-path instance sampling[2]. To further capture both the semantics and structural correlations between different types of nodes, we adopt Convolution Neural Network (CNN) [38] parameterized by Θ to transform ρ of lengths l into meta-path embedding as

$$\mathbf{c}_\mathbf{p} = \text{max-pooling}\left(\left\{CNN\left(\{\mathbf{X}_i^\rho\};\Theta\right)\right\}_{i=1}^{L}\right) \tag{3}$$

where $\{\mathbf{X}_i^\rho\}$ denote the set of embeddings for L path instances from meta-path $\mathbf{p}$, where each $\mathbf{X}_i^\rho$ denotes the embedding matrix of a path instance ρ. Assume L path instances can be generated under a meta-path $\mathbf{p}$, we apply max pooling operation [40] to aggregate them into one embedding $\mathbf{c}_\mathbf{p}$.

We then learn the interaction-specific meta-path context representation. Having obtained the meta-path embeddings $\mathbf{c}_\mathbf{p}$ of meta-path $\mathbf{p}$, we pair the meta-path embedding $\mathbf{c}_\mathbf{p}$ with the current user state s_t and a dispensing action a_t. The dispensing action a_t can be represented as a one-hot representation $\mathbf{a}_t$ overall potential actions in $\mathcal{A}$, however, such a one-hot encoding manner may result in high computation complexity. Thus, we implement a simple embedding lookup layer to transform the one-hot representation $\mathbf{a}_t$ of action a_t into low-dimensional dense vectors:

$$\mathbf{e}_t = \mathrm{Q}^\top \cdot \mathbf{a}_t \tag{4}$$

where $\mathrm{Q}^\top \in \mathbb{R}^{|\mathcal{A}|\times d}$ is the parameter matrix which stores the latent factors of actions and d is the embedding dimension, $\mathbf{e}_t \in \mathbb{R}^d$ is the dense embedding of action a_t. Given user state s_t , context embedding $\mathbf{c}_\mathbf{p}$ and action embedding $\mathbf{e}_t$, we implement a two-layer attention mechanism as

$$\boldsymbol{\alpha}^{(1)}_{s_t,a_t,\mathbf{p}} = f\left(\mathbf{W}_u^{(1)} s_t + \mathbf{W}_a^{(1)}\mathbf{e}_t + \mathbf{W}_\mathbf{p}^{(1)}\mathbf{c}_\mathbf{p} + \mathbf{b}^{(1)}\right) \tag{5}$$

$$\boldsymbol{\alpha}^{(2)}_{s_t,a_t,\mathbf{p}} = f\left(\mathbf{W}^{(2)^\top}\boldsymbol{\alpha}^{(1)}_{s_t,a_t,\mathbf{p}} + \mathbf{b}^{(2)}\right) \tag{6}$$

[2]Details of *Meta-path Based Random Walks* can be found in Appendix A.2.

where $\{\mathbf{W}^{(1)}\}$ and $\mathbf{b}^{(1)}$ denote the weight matrix and the bias vector for the first layer, and the $\mathbf{W}^{(2)}$ and $b^{(2)}$ denote the weight vector and the bias for the second layer, $f(\cdot)$ is set to the ReLU function. As we care about the user-action interaction, we select all meta-path schemes whose starting node type is *user* while the ending node type is *action* (i.e., item), and denote those meta-path schemes into meta-path schemes set $\mathcal{M}_{u\to a}$. We then normalize the attentive scores $\boldsymbol{\alpha}^{(2)}_{s_t,a_t,\mathbf{p}}$ in Eq. (6) over all meta-path schemes in $\mathcal{M}_{u\to a}$ using a softmax function, and derive the final interaction-specific meta-path context representation $\mathbf{c}_{u\to a}\in\mathbb{R}^d$ as a weighted sum:

$$\mathbf{c}_{u\to a}=\sum_{\mathbf{p}\in\mathcal{M}_{u\to a}}\frac{\exp\left(\boldsymbol{\alpha}^{(2)}_{s_t,a_t,\mathbf{p}}\right)}{\sum_{\mathbf{p}'\in\mathcal{M}_{u\to a}}\exp\left(\boldsymbol{\alpha}^{(2)}_{s_t,a_t,\mathbf{p}'}\right)}\cdot\mathbf{c}_{\mathbf{p}} \tag{7}$$

4.1.3 Target Policy Learning. For each time $t\in T$, the interaction-specific meta-path context representation can provide important semantics to regulate the user state and the involved actions. Therefore, user state s_t and action representation $\mathbf{e}_t$ should be adjusted accordingly based on context representation $\mathbf{c}_{u\to a}$ for the later off-policy learning. Specifically, we first compute the attention vectors of meta-path context on users state and actions in *<user state - meta-path context - action>* pairs, then use these attention vectors to refine the user state/action representations in origin space. Formally, giving user state s_t in Eq. (2) and the action representation $\mathbf{e}_t$ in Eq. (4), and the meta-path based context embedding $\mathbf{c}_{u\to a}$ connecting them, we use a single-layer network to compute the attention vectors $\boldsymbol{\beta}^u$ and $\boldsymbol{\beta}^a$ for user state s_t and action a_t as

$$\begin{aligned}\boldsymbol{\beta}^u_t&=Relu\left(\mathbf{W}_u s_t+\mathbf{W}_{u\to a}\mathbf{c}_{u\to a}+\mathbf{b}_u\right)\\ \boldsymbol{\beta}^a_t&=Relu\left(\mathbf{W}_a \mathbf{e}_t+\mathbf{W}_{u\to a}\mathbf{c}_{u\to a}+\mathbf{b}_a\right)\end{aligned} \tag{8}$$

where $\mathbf{W}_u$ and $\mathbf{b}_u$ denote the weight matrix and bias vector for user state attention; $\mathbf{W}_a$ and $\mathbf{b}_a$ denote the weight matrix and bias vector for action attention. Then, the final representations of user states and actions are computed by using an element-wise product with the attention vectors:

$$\begin{aligned}\tilde{s}_t&=\boldsymbol{\beta}^u_t\odot s_t\\ \tilde{\mathbf{e}}_t&=\boldsymbol{\beta}^a_t\odot \mathbf{e}_t\end{aligned} \tag{9}$$

We now transform the refined representations of user state $\tilde{s}_t$ and action $\tilde{\mathbf{e}}_t$, along with interaction-specific context representations $\mathbf{c}_{u\to a}$ into the HIN-enhanced state $s^{u\to a}_t\in\mathbb{R}^d$, to parametrize the policy π_θ. Specifically, the three embedding vectors are combined into a unified representation at the current interaction $t\in T$ as

$$s^{u\to a}_t=\tilde{s}_t\oplus\mathbf{c}_{u\to a}\oplus\tilde{\mathbf{e}}_t \tag{10}$$

where $\oplus$ denotes the vector concatenation operation. The $s^{u\to a}_t$ serves as the final representation of the HIN-enhanced user interests taken at time t. The policy $\pi_\theta(a_t|s^{u\to a}_t)$, which is the probability of recommending the action a_t given the possible action space $\mathcal{A}_t$, is modeled as a softmax function:

$$\pi_\theta(a_t\mid s^{u\to a}_t)=\frac{\exp\left(\mathbf{e}^\top_{t+1}s^{u\to a}_t\right)}{\sum_{a_t\in\mathcal{A}_t}\exp\left(\mathbf{e}_t{}^\top s^{u\to a}_t\right)} \tag{11}$$

where $\mathbf{e}_{t+1}\in\mathbb{R}^d$ is also derived by the embedding lookup operation as denoted in Eq. (4). In fact, $\mathbf{e}_{t+1}$ is the embedding of action a_{t+1} which dispensed in the next time step $t+1$.

4.2 Counterfactual Risk Minimization for Recommendation

The goal of off-policy learning is to maximize each user's long term satisfaction with the system, given the historical logged data generated by historical logging policy π_0. Remember that the target policy π_θ, which is what we care about most, serves to optimize RS to maximize the objective cumulative rewards in (1). To achieve this goal, we have to address the counterfactual question that how much reward would be received if a new target policy π_θ in Eq. (11) had been deployed instead of the original logging policy π_0. This counterfactual question is not easy to address, since the target policy π_θ is different from the historical logging policy π_0 in the off-policy setting[20, 32, 61]. Here we apply *Counterfactual Risk Minimization* (CRM) [49] to correct the discrepancy between the target policy π_θ and logging policy π_0, thus to answer the counterfactual question.

It is well-known that Inverse Propensity Scoring (IPS) estimator is a common practice to correct the discrepancy between π_θ and π_0 [4, 54, 62]. However, the IPS estimator suffers from "propensity overfitting" issue due to the uncertainty on rare actions [13, 30, 39]; when directly optimizing IPS within a learning algorithm, the results tend to have a large variance. To reduce the variance, we resort clipped estimator that caps the propensity ratios (i.e., importance weight) to a maximum value [4]. The core idea is to regulate large weights necessarily associated with actions that are different to the logging policy. The clipped estimator (cIPS) can be represented as

$$L_{\text{cIPS}}(\pi_\theta)=\frac{1}{T}\sum_{t=1}^{T}r_t\min\left\{\frac{\pi_\theta(a_t\mid s^{u\to a}_t)}{\pi_0(a_t\mid s^{u\to a}_t)},c\right\} \tag{12}$$

where c is a constant that serves as the regulator for constraining the importance weight $\frac{\pi_\theta(a_t|s_t)}{\pi_0(a_t|s_t)}$ to at most c, smaller value of constant c reduces variance in the gradient estimate, but introduces larger bias. We thus follow Joachims et.al [31] to prevent the additional bias by adding an empirical variance penalty term λ as

$$L^\lambda_{\text{cIPS}}(\pi_\theta)=\frac{1}{T}\sum_{t=1}^{T}(r_t-\lambda_t)\left\{\frac{\pi_\theta\left(a_t\mid s^{u\to a}_t\right)}{\pi_0\left(a_t\mid s^{u\to a}_t\right)},c\right\} \tag{13}$$

where λ_i regulates the corresponding reward r_i at each interaction By plugging Eq. (13) into objective function Eq. (1), we have:

$$\begin{aligned}R(\pi_\theta)&=\mathbb{E}_{\pi_\theta}\left[\gamma^t L^\lambda_{\text{cIPS}}(\pi_\theta)\right]\\ &=\mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T}\gamma^t\left(r(s^{u\to a}_t,a_t)-\lambda_t\right)\left\{\frac{\pi_\theta\left(a_t\mid s^{u\to a}_t\right)}{\pi_0\left(a_t\mid s^{u\to a}_t\right)},c\right\}\right]\end{aligned} \tag{14}$$

4.3 Top-K recommendation

In our experiment, we focus on the widely-adopted Top-K recommendation task, and utilize the two-stage policy gradient strategy [39] as our learning method. The two-stage setup with candidate generation followed by ranking has been widely adopted in industry [3, 14, 18], which is capable of recommending highly personalized items from a huge item space in real-time. In the training phrase, the trained target policy π_θ in Eq. (14) is fed into candidate generation model to form the probability over the possible candidate sets $\mathcal{A}_t\in\mathcal{A}$ conditioning on the current state s_t, denoted by $p_\theta\left(\mathcal{A}_t\mid s_t\right)$. The possible candidate sets $\mathcal{A}_t$ can be the combination of any items $i\in\mathcal{I}$. 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 $q_\theta\left(a_t \mid s_t, \mathcal{A}_t\right)$. Assuming that the policy takes a function form π_θ parameterized by $\theta \in \mathbb{R}^d$, the policy gradient of the cumulative reward function Eq. (14) w.r.t. θ can be expressed as the following REINFORCE gradient thanks to the log-trick:

$$\begin{aligned}\nabla_\theta R\left(\pi_\theta\right) &= \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} \gamma^t \left(r(s_t^{u\to a}, a_t) - \lambda_t\right) \nabla_\theta \log \pi_\theta(a_t \mid s_t^{u\to a}) \left\{\frac{\pi_\theta(a_t \mid s_t^{u\to a})}{\pi_0(a_t \mid s_t^{u\to a})}, c\right\}\right] \\ &= \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T} \gamma^t \left(r(s_t^{u\to a}, a_t) - \lambda_t\right) \left\{\frac{\sum_{\mathcal{A}_t} q_\theta\left(a_t \mid s_t^{u\to a}, \mathcal{A}_t\right) \nabla_\theta p_\theta\left(\mathcal{A}_t \mid s_t^{u\to a}\right)}{\pi_0(a_t \mid s_t^{u\to a})}, c\right\}\right]\end{aligned} \tag{15}$$

Here, we consider REINFORCE [55], a typical policy gradient method, as our optimization algorithm is to search the optimal policy in recommendation.

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 HINpolicy perform compared with state-of-the-art off-policy recommendation methods?
- **RQ2:** How does HIN information affect our method and different sparsity levels of user feedback?
- **RQ3:** How do hyper-parameters in our method impact the recommendation performance?

5.1 Experimental Setup

5.1.1 Logging Policy from Logged Data. We adopt two widely used public datasets from different domains, namely MovieLens and Douban-book. For both datasets, we binarize the feedback data (i.e., ratings) by interpreting ratings of 4 or higher as the positive feedback (i.e., $r = 1$), otherwise negative (i.e., $r = 0$). The detailed statistics of all datasets are given in Table 1. To facilitate the utility of real-world recommendation datasets in off-policy learning, we start with designing simulation experiments based on an online simulator [9, 63, 64] to recover the missing reward r in partially-observed recommendation datasets. Details of our online simulator are given in Appendix B.1. The logged feedback samples are then acquired by running a logging policy π_0 on the recovered datasets. We adopt the wildly used logging policy, i.e., the uniform-based logging policy. Details of the logging policy can be found in Appendix B.2.

5.1.2 Contextual Information. We consider two publicly available recommendation datasets i.e., MovieLens and Douban-book, and report the selected meta-paths in Table 5 of Appendix B.3. These two datasets contain multiple attributes for both users and items, thus can provide rich contextual information for off-policy learning. We only select short meta-paths of at most four steps, since long meta-paths are likely to contain noise. The three used datasets, i.e.,

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-100K	943	1682	100000	106.0445	59.4530 %	6.30%
MovieLens-1M	6040	3952	1000209	165.5975	253.0893 %	3.95%
Douban Book	13024	22347	792062	60.8156	35.4438 %	0.27%

MovieLens-100K, MovieLens-1M and Douban-book, are produced with diminishing density of 6.30%, 3.95% and 0.27%, respectively, for testing HINpolicy's capability in dealing with the data sparsity.

5.1.3 Baselines. Existing off-policy learning approaches can be roughly sorted into (I) value-based approaches and (II) policy-based approaches. The former generates a target policy based on modeling the actual reward that is received by a certain action, while the latter directly models the actions that should be taken in order to maximise the total cumulative reward a policy will collect. Note that our method can be categorized as a policy-based approach. We perform our method and three representative methods from these two categories on Top-K recommendation task. We evaluate all baselines on the same logged user feedback samples as in our HINpolicy.

- *Bandit-MLE* [26](I): is a value-based approach that estimates the likely reward (i.e., value) a certain action would yield through Maximum Likelihood Estimation (MLE), then generates target policy by selecting actions that have the maximum value. It applies the IPS estimator to adjust for the difference in the distribution of logging policy and target policy to eliminate bias.
- *POEM* [49] (II): Optimizer for Exponential Models (POEM) is the earliest policy-based approach that uses IPS-based counterfactual risk minimisation for off-policy bias correction.
- *BanditNet* [31] (II): is a recent notable extension of off-policy learning with logged user feedback using counterfactual risk minimisation, it optimises an additional Lagrangian form of SNIPS estimator and extends the off-policy learning to deep neural networks.

We evaluate all baseline methods using Precision@K and Normalized Discounted Cumulative Gain (NDCG)@K with $K = [1, 5, 10, 20]$. The implementation details are given in Appendix B.4.

5.2 Performance Comparison (RQ1)

Table 2 reports the experimental results by an average of 5 repeated experiment runs of evaluation. Note that both our method and all baselines are performed under the uniform-based logging policy. The uniform-based logging policy samples every action at random with an equal probability, which is an idealised (i.e., unbiased) setting that a recommender desire. Analyzing Table 2, we have the following observations:

- Our proposed HINpolicy consistently yields the best performance among all datasets on both evaluation metrics. By averaging the performance under all K, our method achieves significant improvements over the best baseline. Particularly, our method improves Precision@K by 25.40%, 5.85%, 38.18% and NDCG@K by 27.47%, 6.38%, 41.78% on MovieLens-100K,

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

Datasets	Metrics	Bandit-MLE	BanditNet	POEM	HINpolicy	Improv.%
MovieLens-100k	Precision@1	0.705	0.671	0.709	**0.894**	26.09%
	Precision@5	0.673	0.663	0.692	**0.862**	24.57%
	Precision@10	0.652	0.645	0.666	**0.832**	24.92%
	Precision@20	0.649	0.622	0.642	**0.818**	26.04%
	NDCG@1	0.692	0.683	0.684	**0.801**	15.75%
	NDCG@5	0.678	0.667	0.661	**0.872**	28.61%
	NDCG@10	0.649	0.621	0.639	**0.848**	30.66%
	NDCG@20	0.627	0.605	0.631	**0.851**	34.87%
MovieLens-1M	Precision@1	0.789	0.727	0.781	**0.883**	11.91%
	Precision@5	0.791	0.811	0.857	**0.931**	8.63%
	Precision@10	0.777	0.867	0.901	**0.925**	2.66%
	Precision@20	0.755	0.888	0.917	**0.919**	0.22%
	NDCG@1	0.741	0.736	0.759	**0.852**	12.25%
	NDCG@5	0.822	0.748	0.821	**0.898**	9.25%
	NDCG@10	0.871	0.822	0.828	**0.896**	2.87%
	NDCG@20	0.908	0.851	0.931	**0.942**	1.18%
Douban Book	Precision@1	0.659	0.625	0.611	**0.943**	43.10%
	Precision@5	0.649	0.616	0.593	**0.875**	34.82%
	Precision@10	0.626	0.602	0.568	**0.852**	36.10%
	Precision@20	0.599	0.572	0.545	**0.831**	38.73%
	NDCG@1	0.608	0.582	0.628	**0.817**	30.10%
	NDCG@5	0.588	0.567	0.612	**0.889**	45.26%
	NDCG@10	0.553	0.548	0.594	**0.868**	46.13%
	NDCG@20	0.549	0.528	0.572	**0.833**	45.63%

MovieLens-1M and Douban Book, respectively. This indicates that HINpolicy indeed improves the Top-K recommendation thanks to the modeling of rich context information.

- Across the datasets, all baseline models achieve downgraded performance on MovieLens-100K compared with those on MovieLens-1M. We consider this is because the size of the former is much smaller than the latter, while the small dataset size easily renders sub-optimal learning of the recommendation policy. However, our HINpolicy can still achieve satisfactory results and adapts well with limited samples, since the HIN provides the auxiliary information to augment the off-policy learning. Meanwhile, the degrade can be also found for the performance of baselines on Douban Book and MovieLens-1M. This is because Douban-book is much sparser than MovieLens-1M, with 0.27% and 4.19% density, respectively. Likewise to MovieLens-1M, HINpolicy outperforms all baselines on Douban Book. This indicates that HINpolicy handles the sparsity well with the support of rich side knowledge.
- Across the baseline models, policy-based POEM outperforms value-based Bandit-MLE in most cases, since it learns policy directly through calculating the end-to-end cumulative rewards without incorporating another decision-making stage like value-based approaches. Moreover, considering the IPS estimator used in POEM, although BanditNet contains an advanced SNIPS estimator, it still cannot outperform POEM. We infer that regulating the inverse weights with SNIPS is not well suited in the recommendation task. This is mainly because SNIPS introduces multiplicative control variate to the IPS estimator that heavily penalises the target policy, limiting its exploration ability of policy learning in the recommendation. On the contrary, we consider leveraging the IPS-based clipped estimator with penalty term, which can reduce bias and variance effectively.

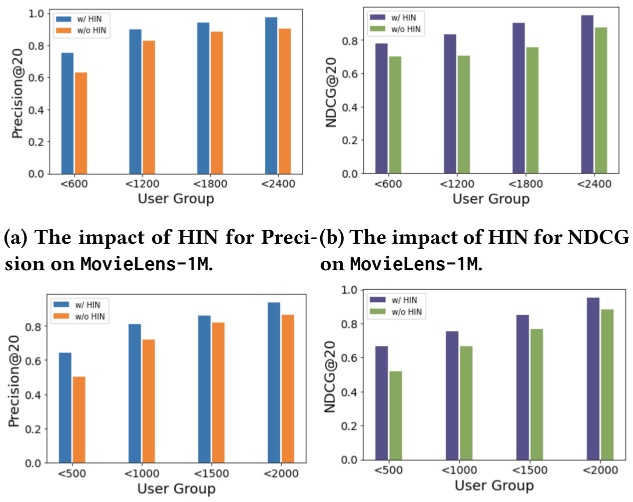
5.3 Study of HIN (RQ2)

To thoroughly investigate HINpolicy's contributions on alleviating the sparsity issue in HIN information, we conduct how the HIN assist in achieving better recommendation in presence of various sparsity of logged data. To characterize the sparsity levels, we divide users into four groups based on their total amounts of interactions with items. For example, in MovieLens-1M, user group 1 represents users who have less than 600 ratings for movies; likewise user group 2, 3 and 4 have less than 1200, 1800 and 2400 ratings, respectively. Hence, user group 1 represents the sparsest data level, while user group 4 represents the densest data level. Table 3 shows the overall performance of our HINpolicy with or without HIN, based on 5 runs of repeated training on MovieLens-1M and Douban Book. With the trained HINpolicy-w/ HIN and HINpolicy-w/o HIN model, we give the test results of the two models on the four user groups and show the comparisons in Figure 4. We have the following observations: 1) Apparently, HINpolicy augmented with HIN outperforms the counterpart without HIN, evidences can be found in both Table 3 and Figure 4. Moreover, facing different user groups as denoted in Figure 4, the test recommendation performance of HINpolicy-w/ HIN consistently outperforms HINpolicy-w/o HIN. These promising discoveries confirm HIN has significant effects on policy learning, which can benefit to achieve satisfying recommendations; 2) HIN information has a critical effect on sparse dataset Douban Book, especially on the sparsest user group (i.e., ratings <500), with the precision of 0.642% and 0.501% and the NDCG of 0.667% and 0.521% for HINpolicy-w/ HIN and HINpolicy-w/o HIN, respectively. Compared to MovieLens-1M having a higher density, our improvement is not as significant as in the Douban Book having a lower density. Hence, we conclude that HINpolicy has a more significant effect on the sparse dataset.

Table 3: Our performance with (w/) or without (w/o) HIN.

Dataset	Metrics	HINpolicy-w/ HIN	HINpolicy-w/o HIN
MovieLens-1M	Precision@20	0.918	0.730
	NDCG@20	0.942	0.752
Douban Book	Precision@20	0.835	0.708
	NDCG@20	0.831	0.720

In a nutshell, incorporating HIN can significantly improve recommendation performance in off-policy setting, while advantages of HIN are more apparent when the dataset is extremely sparse.



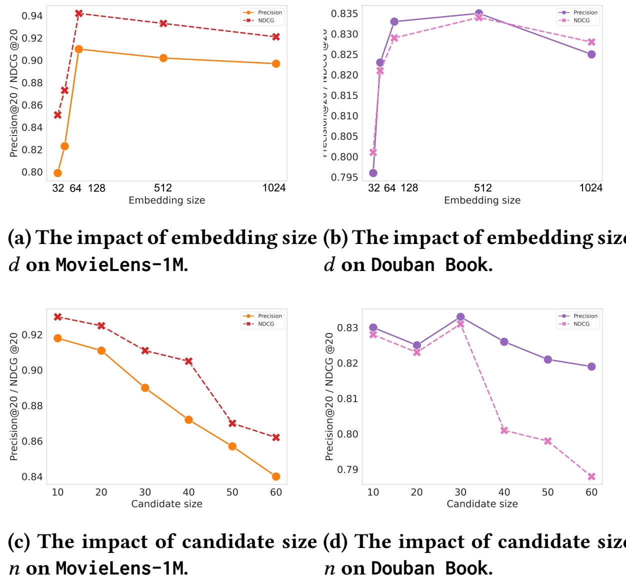
(a) The impact of HIN for Precision on MovieLens-1M.
(b) The impact of HIN for NDCG on MovieLens-1M.
(c) The impact of HIN for Precision on Douban Book.
(d) The impact of HIN for NDCG on Douban Book.

Figure 4: Effectiveness analysis on HIN: different user groups control the interaction numbers.

5.4 Case Study (RQ3)

A case study on HINpolicy is also conducted to investigate the impact of parameters on the model performance. Specifically, our method has two key parameters, i.e., embedding size d that controls the latent factor numbers of user state, action and context representations; candidate length n that controls the length of candidate generation set that waits to be re-ranked by the ranking. For all parameters listed above, we vary the value of one parameter while keeping the others unchanged. Figure 5 shows the parameter sensitivities of d and n for Precision@20 and NDCG@20 on both MovieLens-1M and Douban Book datasets. From both sub-figures, it is easy to see that relative trends of model performance variations are consistent across different datasets.

- For the embedding dimension d, in Figure 5 (a) (b), the recommendation results on MovieLens-1M and Douban Book achieve the peak when $d = 128$, then tend to be stable afterwards. We consider the witness of the increasing trend from $d = 32$ to $d = 128$ is reasonable: as d controls the latent vectors of state/action/context embeddings, exiguous (say $d < 128$) latent factors can not retain sufficient information, thus can not serve well the later policy learning. The stable performance of our HINpolicy after d is set to 128 demonstrates that our model is robust towards varying embedding dimensions.
- For candidate generation length n, in Figure 5 (c) (d), we find that the model performance on both datasets decrease when n increases from 10 to 40, while the impact of n is more severe on NDCG@20 compared with it on Precision@20. The candidate generation length n in the two-stage setting we adopt can be interpreted as the number of candidate actions that wait to be re-ranked by the ranking model. Thus, when n increases, it is harder for the ranking model to give correct results, resulting in the shown decrease trends. We consider this as the “curse” of the two-stage setup with candidate generation followed by ranking, however, we value the two-stage setup for its ability to recommend relevant items from a huge space in real-time, which is suitable for training massive logged user feedback data. We leave the exploration of a more robust ranking model for our future work.



(a) The impact of embedding size d on MovieLens-1M.
(b) The impact of embedding size d on Douban Book.
(c) The impact of candidate size n on MovieLens-1M.
(d) The impact of candidate size n on Douban Book.

Figure 5: Parameter sensitiveness: embedding size d controls the latent factor numbers of user, action and context embeddings; Candidate length n controls the length of candidate generation set.

6 CONCLUSION AND FUTURE WORK

In this paper, we have researched the bias in logged user feedback data, and proposed the first principled approach HINpolicy to achieve unbiased off-policy learning for the recommendation. Our proposed HINpolicy is capable of generating high-quality recommendation policy in virtue of the informative knowledge over the given HIN. In addition, counterfactual risk minimization adaptively corrects the rewards and produces the unbiased recommendation policy that maximizes users' long-term satisfaction. We evaluate our HINpolicy on three real-world recommendation datasets, with extensive experiments and in-depth analyses demonstrating its' robustness and effectiveness. In future work, we are interested in designing a robust ranking model which can adapt with dynamic searching space for a two-stage policy gradient.

REFERENCES

[1] M Mehdi Afsar, Trafford Crump, and Behrouz Far. 2021. Reinforcement learning based recommender systems: A survey. *arXiv preprint arXiv:2101.06286* (2021).

[2] Ting Bai, Lixin Zou, Wayne Xin Zhao, Pan Du, Weidong Liu, Jian-Yun Nie, and Ji-Rong Wen. 2019. CTrec: A long-short demands evolution model for continuous-time recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 675–684.

[3] Fedor Borisyuk, Krishnaram Kenthapadi, David Stein, and Bo Zhao. 2016. CaSMoS: A framework for learning candidate selection models over structured queries and documents. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 441–450.

[4] Léon Bottou, Jonas Peters, Joaquin Quiñonero-Candela, Denis X Charles, D Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Simard, and Ed Snelson. 2013. Counterfactual Reasoning and Learning Systems: The Example of Computational Advertising. *Journal of Machine Learning Research* 14, 11 (2013).

[5] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and debias in recommender system: A survey and future directions. *arXiv preprint arXiv:2010.03240* (2020).

[6] 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.

[7] Weijian Chen, Yulong Gu, Zhaochun Ren, Xiangnan He, Hongtao Xie, Tong Guo, Dawei Yin, and Yongdong Zhang. 2019. Semi-supervised User Profiling with Heterogeneous Graph Attention Networks.. In *IJCAI*, Vol. 19. 2116–2122.

[8] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *Proceedings of the eleventh ACM international conference on web search and data mining.* 108–116.

[9] 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).

[10] 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.

[11] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems.* 7–10.

[12] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[13] Amanda Coston, Alan Mishler, Edward H Kennedy, and Alexandra Chouldechova. 2020. Counterfactual risk assessments, evaluation, and fairness. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency.* 582–593.

[14] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems.* 191–198.

[15] 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.

[16] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining.* 135–144.

[17] 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).

[18] Chantat Eksombatchai, Pranav Jindal, Jerry Zitao Liu, Yuchen Liu, Rahul Sharma, Charles Sugnet, Mark Ulrich, and Jure Leskovec. 2018. Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In *Proceedings of the 2018 world wide web conference.* 1775–1784.

[19] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. 2018. More robust doubly robust off-policy evaluation. In *International Conference on Machine Learning.* PMLR, 1447–1456.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] Alex Graves. 2012. Long short-term memory. In *Supervised sequence labelling with recurrent neural networks.* Springer, 37–45.

[25] Yulong Gu, Zhuoye Ding, Shuaiqiang Wang, and Dawei Yin. 2020. Hierarchical User Profiling for E-commerce Recommender Systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining.* 223–231.

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

[27] 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.

[28] Chao Huang, Xian Wu, Xuchao Zhang, Chuxu Zhang, Jiashu Zhao, Dawei Yin, and Nitesh V Chawla. 2019. Online purchase prediction via multi-scale modeling of behavior dynamics. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* 2613–2622.

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

[30] Olivier Jeunen, David Rohde, Flavian Vasile, and Martin Bompaire. 2020. Joint Policy-Value Learning for Recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.* 1223–1233.

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

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

[33] 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.

[34] Yu Lei, Hongbin Pei, Hanqi Yan, and Wenjie Li. 2020. Reinforcement learning based recommendation with graph convolutional q-network. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 1757–1760.

[35] 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.

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

[37] Feng Liu, Ruiming Tang, Xutao Li, Weinan Zhang, Yunming Ye, Haokun Chen, Huifeng Guo, Yuzhou Zhang, and Xiuqiang He. 2020. State representation modeling for deep reinforcement learning based recommendation. *Knowledge-Based Systems* 205 (2020), 106170.

[38] Shih-Chung B Lo, Heang-Ping Chan, Jyh-Shyan Lin, Huai Li, Matthew T Freedman, and Seong K Mun. 1995. Artificial convolution neural network for medical image pattern recognition. *Neural networks* 8, 7-8 (1995), 1201–1214.

[39] 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.

[40] Naila Murray and Florent Perronnin. 2014. Generalized max pooling. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2473–2480.

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

[42] Hanh TH Nguyen, Martin Wistuba, Josif Grabocka, Lucas Rego Drumond, and Lars Schmidt-Thieme. 2017. Personalized deep learning for tag recommendation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining.* Springer, 186–197.

[43] 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.

[44] Otmane Sakhi, Stephen Bonner, David Rohde, and Flavian Vasile. 2020. BLOB: A Probabilistic Model for Recommendation that Combines Organic and Bandit

Signals. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 783–793.

[45] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing* 45, 11 (1997), 2673–2681.

[46] Jieun Son and Seoung Bum Kim. 2017. Content-based filtering for recommendation systems using multiattribute networks. *Expert Systems with Applications* 89 (2017), 404–412.

[47] Yi Su, Maria Dimakopoulou, Akshay Krishnamurthy, and Miroslav Dudík. 2020. Doubly robust off-policy evaluation with shrinkage. In *International Conference on Machine Learning*. PMLR, 9167–9176.

[48] 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.

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

[50] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 565–573.

[51] Philip Thomas and Emma Brunskill. 2016. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*. PMLR, 2139–2148.

[52] 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.

[53] 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.

[54] 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.

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

[56] Yao Wu, Christopher DuBois, Alice X Zheng, and Martin Ester. 2016. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the ninth ACM international conference on web search and data mining*. 153–162.

[57] 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*.

[58] Ruobing Xie, Shaoliang Zhang, Rui Wang, Feng Xia, and Leyu Lin. 2021. Hierarchical Reinforcement Learning for Integrated Recommendation. In *Proceedings of AAAI*.

[59] Xin Xin, Alexandros Karatzoglou, Ioannis Arapakis, and Joemon M Jose. 2020. Self-supervised reinforcement learning for recommender systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 931–940.

[60] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. 2019. A simple convolutional generative network for next item recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 582–590.

[61] Houssam Zenati, Alberto Bietti, Matthieu Martin, Eustache Diemert, and Julien Mairal. 2020. Counterfactual learning of continuous stochastic policies. *arXiv preprint arXiv:2004.11722* (2020).

[62] Houssam Zenati, Alberto Bietti, Matthieu Martin, Eustache Diemert, and Julien Mairal. 2021. Counterfactual Learning of Stochastic Policies with Continuous Actions: from Models to Offline Evaluation. (2021).

[63] 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.

[64] 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.

A METHODOLOGY SUPPLEMENTARY

A.1 Notions and Definitions

The important notions we use throughout the methodology is given in Table 4.

Table 4: Key notations and descriptions.

Notation	Description
π_θ	target recommendation policy
π_0	users' behaviour policy (logging policy)
θ	model parameter
T	the total cumulative time step
$\mathbf{p}$	a meta-path scheme
ρ	a meta-path node sequence under meta-path scheme $\mathbf{p}$
o_t	user's interacted items list
s_t	user state representation
$\mathbf{a}_t$	one-hot encoding of action a_t
$\mathbf{e}_t$	dense representation of action a_t
$\mathbf{c}_\mathbf{p}$	meta-path embedding of $\mathbf{p}$
$c_{u\to a}$	interaction-specific context representation
$\tilde{s}_t$	context-aware user state representation
$\tilde{\mathbf{e}}_t$	context-aware action state representation
$s_t^{u\to a}$	HIN-enhanced state representation

Heterogeneous Information Network and Meta-path can be defined using the following paradigms:

Definition 1 (Heterogeneous Information Network). *A Heterogeneous information network is defined as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a node type mapping function: $\phi : \mathcal{V} \to \mathcal{K}$ and an edge type mapping function: $\psi : \mathcal{E} \to \mathcal{J}$, where $\mathcal{K}$ and $\mathcal{J}$ are the node type set and edge type set of $\mathcal{G}$, respectively. Each node $v \in \mathcal{V}$ and edge $e \in \mathcal{E}$ in a HIN belong to one particular type $\phi(v) \in \mathcal{K}$ and $\psi(e) \in \mathcal{J}$, where $|\mathcal{K}| + |\mathcal{J}| > 2$.*

Definition 2 (Meta-path). *Meta-path $\mathbf{p}$ is a path defined on the network schema $T_\mathcal{G} = (\mathcal{K}, \mathcal{J})$, and is denoted in the form of*

$$\mathbf{p} \triangleq (\mathcal{K}_1 \xrightarrow{\mathcal{J}_1} \mathcal{K}_2 \xrightarrow{\mathcal{J}_2} \ldots \xrightarrow{\mathcal{J}_l} \mathcal{K}_{l+1})$$

which defines a composite $\mathcal{J} = \mathcal{J}_1 \circ \mathcal{J}_2 \circ \ldots \circ \mathcal{J}_l$ between type $\mathcal{K}_1$ and $\mathcal{K}_{l+1}$, where $\circ$ denotes the composition operator on relations. For simplicity, we use node type names to denote the corresponding meta-path if no multiple relations exist between type pairs, as $\mathbf{p} = (\mathcal{K}_1\mathcal{K}_2 \ldots \mathcal{K}_{l+1})$.

A.2 Path Sampling

The basic idea of *Meta-path Based Random Walks* [16] is to put random walkers in a HIN to generate paths that constitute multiple types of nodes. Specifically, given $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{K}, \mathcal{J}, \phi, \psi)$, the node sequence $\mathbf{n}_\mathbf{p} = \{v_1, \cdots, v_{i+1}\}$ under a specific meta-path $\mathbf{p}$ is generated according to the following distribution:

$$P(v_{i+1} \mid v_i, \mathbf{p}) = \begin{cases} \frac{1}{\left|\mathcal{N}_{v_i}^{(\mathcal{K}_{l+1})}\right|}, & (v_i, v_{i+1}) \in \mathcal{E} \text{ and } \phi(v_{i+1}) = \mathcal{K}_{l+1} \\ 0, & \text{otherwise} \end{cases} \quad (16)$$

where $\mathcal{N}_{v_i}^{(\mathcal{K}_{l+1})}$ is the first-order neighbor set for node v_i whose type is $\mathcal{K}_{l+1}$; v_{i+1} is the $i+1$-th node whose type is $\mathcal{K}_{l+1}$, and v_i is the i-th node in the walk which belongs to type $\mathcal{K}_l$. By regulating $v_i \in \mathcal{K}_l$ while $v_{i+1} \in \mathcal{K}_{l+1}$, the node types sampled by random walkers is conditioned on the pre-defined meta-path $\mathbf{p}$.

B EXPERIMENTAL SUPPLEMENTARY

B.1 Online Simulator

From the perspective of reinforcement learning, recommendation datasets can be viewed as m records of logged feedback as: $\mathcal{D} = \left\{\left(s^i, a^i, r^i\right)\right\}_{i=1}^m$, where $s^i \in \mathbb{R}^d$ is the feature vector of a user state at data point i, $r^i \in [0,1]$ indicating the user's feedback when action a^i is recommended to the user (e.g., click). Note that the user preference over the whole item corpus is partially observed, i.e., not all user-item pairs are labeled with at least one known user feedback, which is the common nature of recommendation system datasets. As we are interested in the recommendation system, we further adopt the online simulation setting following the common practice in off-policy learning [9, 63, 64]. We build the online simulator h_0 to recover the missing reward r^i conditioned on state s^i an item a^i. The simulator h_0 has a similar model architecture with our HINpolicy, while the output layer is modified as a softmax layer to predict the immediate feedback according to the current state s^i and item a^i. Our well-trained [3] online simulator can allow us to use recommendation system datasets without loss of generality. The off-policy logged data is then acquired by running a logging policy π_0 that samples $r_i \sim \pi_0\left(s^i, a^i\right)$ from $\mathcal{D}$, denoted as $\mathcal{D}_{\text{bandit}} = \left\{\left(s^i, a^i, p^i, r^i\right)\right\}_{i=1}^k$, where k is the total number of samples, p^j is the probability with which that action was taken by the logging policy π_0 (a.k.a propensity), and r^j is the reward. In our model training, we adopt the widely used sampling strategies to define the logging policy π_0 for generating our logged user feedback samples, i.e., uniform sampling for uniform-based logging policy details of the logging policy can be found in Appendix B.2.

B.2 Logging Policy

we adopt the widely used logging policy in our experiment, namely, Uniform-based logging policy. The uniform-based logging policy samples each action at every interaction uniformly at random. It assumes every action's probability of being exposed is $\pi_{\text{uniform}}(a \mid s) = \frac{1}{k}$. This strategy is quite straightforward and free from biases (e.g., popularity bias) [36]. In our experiment, we consider the uniform-based logging policy as an idealised (i.e., unbiased) setting.

B.3 Meta-path in Datasets

The meta-paths we use for our experiments are denoted in the last column of Table 5.

B.4 Parameter Settings

We implement baseline models and our proposed HINpolicy on a Linux server with NVIDIA RTX 3090Ti GPU. For a fair comparison, all logged user feedback samples used for training models are generated through our online simulator together with a logging policy as described in Section 5.1. The logged ratings in `Movielens` and `Douban Book` for training the simulator are split as train/test/validate set with a proportion of 60%/20%/20% of original

[3]With test results of overall 90% precision for the immediate feedback prediction task.

Table 5: Meta-path statistics of the datasets.

Dataset	Node	Meta-path Schemes
MovieLens-100K	#User(U): 943 #Movie(M): 1,682 #Genre(R): 18 #Gender(G): 2 #Age(A): 7 #Occupation(O): 21	UMGM,UMAM, UMRM,UMOM, UMUM
MovieLens-1M	#User(U): 6,040 #Movie(M): 3,952 #Genre(R): 18 #Gender(G): 2 #Age(A): 7 #Occupation(O): 21	UMGM,UMAM, UMRM,UMOM, UMUM
Douban Book	#User(U): 13,024 #Book(B): 22,347 #Group(G): 2,936 #Author(A): 10,805 #Publisher(P): 1,815 #Year(Y): 64 #Location(L): 38	UBGB,UBAB,UBPB, UBYB,UBLB,UBUB

datasets. Without special mention, the final logged user feedback samples are given by applying *uniform* logging policy on the full-information data generated through the trained simulator. As for the policy training, we optimize the two-stage policy gradient with AdaGrad [17], the same gradient descent method is also applied in all the baseline models. A grid search is conducted to choose the optimal parameter combinations in all models considered, with batch size and learning rate searched in $\{32, 64, 128, 512, 1024\}$ and $\{0.0005, 0.001, 0.005, 0.01, 0.05, 0.1\}$, respectively. 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). Unique settings of our HINpolicy are the dimension of HIN embedding vector d, which is searched in $\{8, 16, 32, 64, 128, 256\}$; the number of random walkers for sampling meta-path node sequences are set to default as $n_{walk} = 1000$ with walk length $l_{walk} = 100$; the weight capping constant with $c = 10$. For evaluation, we adopt the two-stage policy gradient method as in [39], different from conventional Learn-to-Rank problems which directly give ranked lists based on sorting the predicted values (e.g., probability) of user-item pairs, we measure the quality of the recommendations given by the ranking model conditioned on the candidate set provided by the candidate generation model. That is, we firstly select top-n items predicted by the candidate generation model as a candidate set, then feed this candidate set to the ranking model to re-rank these candidates. The candidate set length is set as $n = 20$ for all datasets, while we test the final performance on the trained ranking model with ranking length $K = [1, 5, 10, 20]$.