Planning Data Poisoning Attacks on Heterogeneous Recommender Systems in a Multiplayer Setting

Chin-Yuan Yeh[1,3], Hsi-Wen Chen[2], De-Nian Yang[3,4], Wang-Chien Lee[5], Philip S. Yu[6], Ming-Syan Chen[1,2]
[1]*Graduate Institute of Communication Engineering, National Taiwan University, Taiwan*
[2]*Department of Electrical Engineering, National Taiwan University, Taiwan*
[3]*Institute of Information Science, Academia Sinica, Taiwan*
[4]*Research Center for Information Technology Innovation, Academia Sinica, Taiwan*
[5]*Department of Computer Science and Engineering, The Pennsylvania State University, USA*
[6]*Department of Computer Science, University of Illinois Chicago, USA*
{cyyeh, hwchen}@arbor.ee.ntu.edu.tw, dnyang@iis.sinica.edu.tw
wlee@cse.psu.edu, psyu@uic.edu, mschen@ntu.edu.tw

***Abstract*—Data poisoning attacks against recommender systems (RecSys) often assume a single seller as the adversary. However, in reality, there are usually multiple sellers attempting to promote their items through RecSys manipulation. To obtain the best data poisoning plan, it is important for an attacker to anticipate and withstand the actions of his opponents. This work studies the problem of Multiplayer Comprehensive Attack (MCA) from the perspective of the attacker, considering the subsequent attacks by his opponents. In MCA, we target the Heterogeneous RecSys, where user-item interaction records, user social network, and item correlation graph are used for recommendations. To tackle MCA, we present the Multilevel Stackelberg Optimization over Progressive Differentiable Surrogate (MSOPDS). The Multilevel Stackelberg Optimization (MSO) method is used to form the optimum strategies by solving the Stackelberg game equilibrium between the attacker and his opponents, while the Progressive Differentiable Surrogate (PDS) addresses technical challenges in deriving gradients for candidate poisoning actions. Experiments on Heterogeneous RecSys trained with public datasets show that MSOPDS outperforms all examined prior works by up to 10.6% in average predicted ratings and up to 11.4% in HitRate@3 for an item targeted by an attacker facing one opponent. Source code provided in https://github.com/jimmy-academia/MSOPDS**

***Index Terms*—data poisoning attack, recommender system, Stackelberg game, graph neural network**

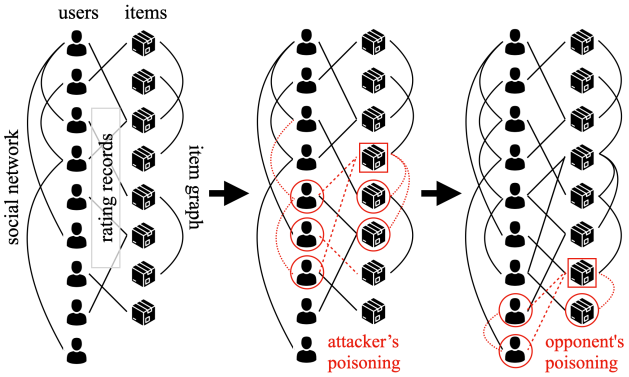


Fig. 1: Illustration of the Multiplayer Comprehensive Attack problem. Red lines indicate poison edges. After an adversary (attacker) poisons the RecSys for his objective, another adversary (opponent) may later poison the RecSys with a different goal. Thus, if the first attacker does not anticipate the *subsequent* opponent's poison actions, his poison effort may be voided by the following opponent's poisons.

I. Introduction

Recommender Systems (RecSys) are used in eCommerce to help users explore and discover items that may be of interest to them. The ranking of items in a RecSys can thus have a significant impact on their visibility and sales [1]. However, since RecSys use past user data to improve their recommendations, users can also collectively affect the behavior of the RecSys through their actions [2]. This opens the door to malicious actors for *data poisoning attacks*, which involve introducing fake or biased data into the system to manipulate its recommendations.

Data poisoning attacks against RecSys have been observed in both research [3] and real-world scenarios. In the context of eCommerce, multiple sellers compete for a shared market, resulting in a complex multiplayer game. Nevertheless, the scenario of multiple adversaries has been largely ignored in previous research. Additionally, prior research has mostly focused on basic RecSys that operate only on user-item interactions (i.e., the rating records) [4]. To address these gaps, we introduce a novel attack scenario called *Multiplayer Comprehensive Attack (MCA)*, where an attacker aims to influence the RecSys while anticipating subsequent *poisoning actions* taken by opponents in a Heterogeneous RecSys environment.

MCA is novel in the following ways: **1) Multiplayer Game.** Existing studies oversimplify the problem by considering only a single attacker [5]. In contrast, as a real eCommerce environment generally consisting of multiple competing sellers [6], we consider a multiplayer game setting where an attacker who generates a batch of poison data may face opponents who also inject data into the RecSys for contradicting objectives, resulting in unexpected or counteractive effects. In our setting (Fig. 1), the attacker must anticipate the subsequent actions of his opponents' while trying to influence the RecSys. **2) Heterogeneous RecSys (Het-RecSys).** To align with real-

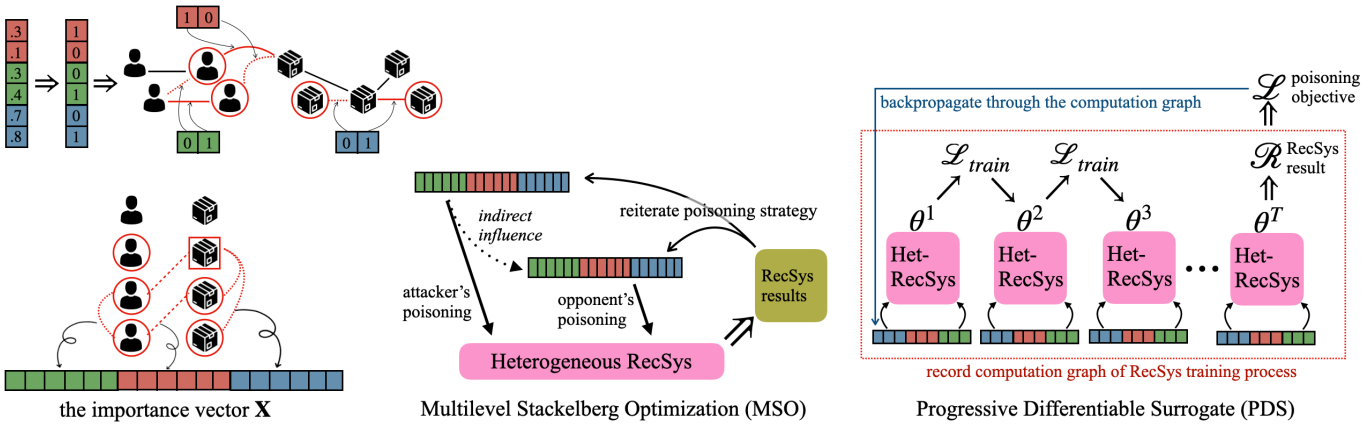


Fig. 2: A comprehensive high-level presentation of the different parts of MSOPDS, which consists of MSO and PDS, and utilizes the importance vector throughout the procedure. **Bottom left:** Each element of the importance vector maps to a candidate poisoning action that may be of different type, e.g., adding poisoning edges in the social network (green) and item graph (blue), or adding poisoning ratings (red). **Upper left:** We obtain a poisoning plan from an importance vector by creating a *binarized* copy of the importance vector, where 1 indicates the top-valued actions (red solid lines) that are selected under budget constraint, and 0 indicates not selected (red dashed lines). **Middle:** Illustration of Multilevel Stackelberg Optimization (MSO). In each iteration, the opponent's vector is updated based on how his poison directly influences his objective. However, the attacker also considers his *indirect influence* on the opponent since the opponent will observe the attacker's poison when deciding his poisoning plan. **Right:** Illustration of Progressive Differentiable Surrogate (PDS), which calculates the partial derivatives of a poisoning objective w.r.t. importance vector by incorporating it into the training process of a Het-RecSys. By recording the computation graph of the training process, the gradient is derived by backpropagation.

world eCommerce platforms, such as Facebook Marketplace, Yelp [7], [8], and Pinterest [9] exploiting social network for recommendation, and Amazon [10] and Netflix [11] exploiting item relations for recommendation we investigate the vulnerability of Het-RecSys [12], which incorporate additional information such as *social network* and *item correlation graph (item graph)* besides the rating records. This is in contrast to prior works which considered basic RecSys using the rating records only [3], [13], [14]. In our work, since Het-RecSys exploit additional information, an attacker also conduct additional computation to obtain the optimal planning of poisoning actions beyond creating fake rating records by hiring real users to manipulate the social network and item graph.[1] Finally, based on common marketing priorities in eCommerce platforms, we focus on promoting a specific item to a targeted audience [22] over a set of *competing items* [23]. Thus, our considerations are overall more "comprehensive" than previous research on data poisoning attacks against RecSys.

To solve the Multiplayer Comprehensive Attack problem, we propose Multilevel Stackelberg Optimization over Progressive Differentiable Surrogate (MSOPDS), which consists of two key components, i.e., Multilevel Stackelberg Optimization (MSO) and Progressive Differentiable Surrogate (PDS), with *importance vectors* representing poisoning plans for attackers. In particular, the importance vector is utilized to record the *priority* of each poisoning action. First, the MSO framework is based on the Stackelberg game which is widely used in eCommerce analyses to models hierarchical relations between multiple players and to effectively exploit the first-mover (leader) advantages over the subsequent opponents (followers) [24].[2] In the MSO framework, the attacker is the *leader* and the subsequent opponents are the *followers*. As shown in the middle of Fig. 2, MSO assists the attacker to anticipate the actions of subsequent opponents and approaches the Stackelberg game equilibrium by accounting for this indirect influence of the attacker on the opponent.

Nevertheless, the partial derivatives required for the update rules of MSO cannot be directly computed due to the nature of Het-RecSys as a Graph Neural Network (GNN) [29]. Specifically, iterating fractional update steps on the poisoning action plans would lead to fractional edge values (esp. in the social network and the item graph) that could not be processed by the Het-RecSys. Note that prior works [3], [13] do not face such a challenge since they plan their data poisoning attacks against basic RecSys which do not utilize graph information.

To circumvent the difficulty, we propose the *Progressive Differentiable Surrogate (PDS)* and utilize the *importance vectors* to represent and record the gradient adjustments on the poisoning plans, where each element represents a candidate poisoning action across the heterogeneous information utilized by Het-RecSys. Intuitively, PDS interfaces between the importance vector and the Het-RecSys. As shown in the upper left of

[1]With the right incentives, eCommerce customers are often encouraged to leave reviews or ratings [15], [16] or connect with social media accounts [17]. Alternatively, real user accounts may also be purchased [18], [19] or hacked [20], [21], giving attackers direct control of those accounts.

[2]Examples include marketing [25], real-time bidding [26], logistics coordination [27], and pricing strategies [28].

Fig. 2, the importance vector is *binarized* and integrated into different parts of the heterogeneous information by the PDS. Then, by recording the training process of the surrogate Het-RecSys with the poisoning actions involved, PDS can compute the derivatives with regards to each poisoning actions required by the update rules of MSO.

The contributions made in this paper are as follows.

- This paper formulates a pioneering problem, namely Multiplayer Comprehensive Attack (MCA), which captures competing data poisoning attacks against RecSys, which is more realistic than the scenarios considered in previous works. To our knowledge, this is not only the first effort to capture multiple competing attacks on a RecSys, but also the first attempt to consider a Het-RecSys, which is more advanced and realistic than the basic RecSys considered in previous works as it exploits user relations in social network as well as item relations in item graph.[3]
- MSOPDS approaches an equilibrium amongst multiple objectives, instead of following the standard gradient optimization that directly optimizes on a single objective.
- New update rules in MSOPDS are designed in accordance with a formal analysis on guaranteed convergence to equilibrium.
- MSOPDS presents Progressive Differentiable Surrogate, a novel GNN-based RecSys design, where poison edges and ratings are separately incorporated into the graph convolution process and the training loss, respectively.
- Extensive experiments on Heterogeneous RecSys trained with real-world datasets show MSOPDS outperforms prior works by up to 10.6% in average predicted ratings and up to 11.4% in HitRate@3 for an item targeted by an attacker facing one opponent, while being less impacted by increase in the capacity or number of opponents.

II. Related Work

The vulnerability of RecSys against data poisoning attacks is well recognized in prior research [3], [5], [13]. In particular, the topic of data poisoning attacks against RecSys [40] is a relevant and important issue to data engineering [40]–[42] since it is increasingly important to have reliable RecSys to enable efficient data access and exploration [43]–[45] in larger amounts of data. However, data poisoning attacks against these systems can undermine the recommendations, leading to financial losses or even societal problems. In a data poisoning attack [46], an attacker *poisons* a machine learning model by injecting adversarial (fake) samples into the training data [47]. While earlier works on poisoning RecSys focus on heuristic approaches (e.g., popular attack or shilling attack [48]), recent works utilize optimization-based methods [3], [13], [14], [40], [49]–[54] to generate poison data that fool the RecSys models under specific requirements. For example, Li et al. [13] apply the Stochastic Gradient Langevin Dynamics to determine the poison data while adhering to the distribution of the rating records. Tang et al. [3] provided an in-depth analysis on the bilevel optimization for data poisoning attack on RecSys. In addition, Zhang et al. [14] tackle the problem of data poisoning attacks using modified and perturbed user-item interaction data. Besides, Song et al. [40] propose to utilize reinforcement learning to decide a poisoning strategy under limited information. Nevertheless, all the aforementioned works target only one single attacker attempting to manipulate the RecSys through data poisoning. In contrast, we formulate the problem of data poisoning attacks as a multiplayer game by considering the effects of the subsequent opponents' poisoning actions.

Furthermore, these works only focus on conducting the poisoning actions with fake user accounts [3], [13] for basic RecSys that only operates on the rating records (e.g., implicit clicks or explicit ratings) [14]. In contrast, we investigate the vulnerability of Graph Neural Network (GNN)-based Het-RecSys [55], [56] in a more realistic scenario with various poisoning actions involving real and fake users as well as modifications on the social networks and the item graph.

On the other hand, there are also other prior studies on test-time attack [29], [57]–[59] and train-time poisoning [60]–[63] against different types of GNNs. For example, Dai et al. [57] formulate a reinforcement-learning based attack method against node classification and graph classification supported by GNN models. Bojchevski and Günnemann [60] poisons the graph topology to manipulate node embeddings learned by graph methods. Besides, Li et al. [58] present a black-box attack on graph learning based community detection models using a novel graph generative neural network. However, the above-mentioned works target different GNNs under a single attacker setting. It is worth noting that our work is the first effort to investigate the GNN-based Het-RecSys under a multiplayer setting.

TABLE I: Notation table.

Symbol	Descriptions
u, i	An individual user or an item.
$\mathcal{U}, \mathcal{I}$	The (real) user set and item set.
$\mathcal{U}^p_{fake}$	The set of fake users created by player p. $\mathcal{U}^p_{fake} \not\subseteq \mathcal{U}$.
$\mathcal{U}^p_{TA}$	The target audience of player p.
$\mathcal{U}^p_{base}$	The customer base of player p.
i^p_t	The target item of player p.
$\mathcal{I}^p_{product}$	The company products of player p.
$\mathcal{I}^p_{compete}$	The competing items of player p.
$\Xi, \varnothing$	The set of rating values $\{1, 2, 3, 4, 5\}$ or missing value.
$\mathbf{R}$	The rating matrix ($\mathbf{R} \in \Gamma^{\lvert\mathcal{U}\rvert \cdot \lvert\mathcal{I}\rvert}, \Gamma \equiv \Xi \cup \{\varnothing\}$).
$\mathcal{G}_U, \mathcal{G}_I$	The social network and item graph ($\mathcal{G} \equiv \mathcal{G}_U \cup \mathcal{G}_I$).
$\hat{\mathbf{R}}, \hat{\mathcal{G}}$	The poisoned ratings and the poisoned graph data.
$\mathcal{R}_{(u,i)}$	The rating predictions (between user u and item i).
$\mathcal{L}^p$	The adversarial loss, i.e., the poisoning objective (of p).
$\mathcal{C}^p$	The capacity set, i.e., candidate poisoning actions (of p).
$\mathcal{X}^p$	The poisoning action plan of player p
$(\hat{\mathbf{X}}^p)$ $\mathbf{X}^p$	The (binarized) importance vector of player p.

[3]Real world incidents [30] suggests that amassing fake reviews to manipulate of RecSys recommendations are attempted by multiple actors [31]–[34] and even with aids from dedicated companies [35], [36]. Besides, the rapidly growing sector of social commerce [37]–[39] indicates the importance of studying Het-RecSys, which can process the heterogeneous information involved.

III. Problem Formulation

For easy reference, the notations are listed in Table I.

A. Heterogeneous RecSys (Het-RecSys)

In general, a RecSys provides a personalized ranking of all items for a given user based on the similarity between their embedding representations learned from past rating records [64]. A *Het-RecSys* [12], [65] extends the basic RecSys by exploiting additional information. Besides the rating records, a Het-RecSys utilizes the social network $\mathcal{G}_U$ and the item graph $\mathcal{G}_I$ to model the social influences among users and associations among items to improve the recommendations.

Definition 1 (**Het-RecSys**). *Given the user set $\mathcal{U}$, item set $\mathcal{I}$, rating matrix $\mathbf{R} \in \Gamma^{|\mathcal{U}|\cdot|\mathcal{I}|}$ with $\Gamma \equiv \Xi \cup \{\varnothing\}$ being a rating out of 5 stars $\Xi \equiv \{1,2,3,4,5\}$ or a missing value $\varnothing$, social network $\mathcal{G}_U = \{\mathcal{U}, \mathcal{E}_U\}$, and item graph $\mathcal{G}_I = \{\mathcal{I}, \mathcal{E}_I\}$, a Het-RecSys predicts the missing rating values in $\mathbf{R}$.*

Utilizing the heterogeneous graph data, Graph Neural Network (GNN)-based Het-RecSys [12] are powerful because they encode the entity features along with the graph topology through graph convolution processes [66]. Formally, let let $\mathcal{R}(\theta, \mathcal{G})$ denote the Het-RecSys prediction result with $\mathcal{G} \equiv \mathcal{G}_U \cup \mathcal{G}_I$ and θ denotes the learnable parameters, the training process aims to minimizes the training loss $\mathcal{L}_{train}$,

$$\min_{\theta} \mathcal{L}_{train}(\mathcal{R}(\theta, \mathcal{G}), \mathbf{R}) + \lambda \|\theta\|_2^2, \tag{1}$$

where $\mathcal{L}_{train}$ is formulated as the Mean Square Error [67] between the predicted ratings $\mathcal{R}$ and the provided ratings $\mathbf{R}$ [12], and λ controls the strength of L_2 regularization [68].

B. Data poisoning attack scenarios

In this work, we investigate the effects of multiple adversaries conducting data poisoning attacks [40] on the same RecSys. Since eCommerce are often open and public [14], each adversary may have access to the full set of records, including poison data injected *prior to his action* by other adversaries. Our goal is thus to understand the potential impact of *subsequent* poisoning attacks on an attacker's data poisoning plan to manipulate a Het-RecSys.

To distinguish the various scenarios considered in the literature and this work, we formally define three different attacks, namely, Injection Attack (IA), Comprehensive Attack (CA), and Multiplayer Comprehensive Attack (MCA). As compared in Table II, while IA targets basic RecSys and CA targets Het-RecSys, they both assume single adversary and are oblivious to the subsequent opponents. On the other hand, MCA targets Het-RecSys and aims to prepare a *resilient* poisoning plan in anticipation of the subsequent opponent actions. Notice that all prior works on data poisoning attack against RecSys [3], [13], [52], [54] can be categorized as IA.

Definition 2 (**Bi-level formulation of data poisoning attack**). *Data poisoning attack (by single adversary) can be formulated as a bi-level optimization [3], where an adversarial loss (i.e., the poisoning objective) $\mathcal{L}$ describing the attacker's objective is optimized in the upper-level, and the training of RecSys over the poisoned data is described in the lower-level.*

$$\begin{aligned} &\min_{\mathcal{X} \subseteq \mathcal{C}} \mathcal{L}(\mathcal{R}(\theta^*, \hat{\mathcal{G}})) \\ &given\ \theta^* \in \arg\min_{\theta} \mathcal{L}_{train}(\mathcal{R}(\theta, \hat{\mathcal{G}}), \hat{\mathbf{R}}), \end{aligned} \tag{2}$$

where $\mathcal{X}$ is the poisoning plan selected (under relevant budget constraints) from the attacker's capacity set $\mathcal{C}$, $\mathcal{R}$ is the RecSys result, while $\hat{\mathbf{R}}$ and $\hat{\mathcal{G}}$ represent the poisoned rating records and the poisoned graph information, respectively.

TABLE II: Comparison of attack scenarios.

	IA	**CA**	**MCA**
targeted RecSys	basic	Het-RecSys	Het-RecSys
# of adversary	1	1	multiple

IA and CA ascribe to the single player assumption and may be formulated following the Bi-level optimization framework by specifying the poisoning objective $\mathcal{L}$ and the capacity $\mathcal{C}$.

Definition 3 (**Injection Attack (IA)**). *An injection attack aims to maximize the rating of a target item i_t to all real users $u \in \mathcal{U}$ by creating fake ratings with a set of fake users $\mathcal{U}_{fake}$. The* Injection Attack loss $\mathcal{L}_{IA}$ *is defined as follows.*

$$\mathcal{L}_{IA} = -\frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathcal{R}_{(u,i_t)}, \tag{3}$$

where subscripts are used in $\mathcal{R}_{(u,i_t)}$ to denote the individual rating prediction between user u and the target item i_t. Following prior works [3], [13], [14], [40], Injection Attack exploits a set of fake users $\mathcal{U}_{fake} \not\subseteq \mathcal{U}$ that are injected into the RecSys. Its capacity set $\mathcal{C}_{IA}$ consists of each fake user giving fake ratings to any items, where a budget constraints restricts how many items each fake user may rate.

$$\mathcal{C}_{IA} = \{(u, i, r) \mid u \in \mathcal{U}_{fake},\ i \in \mathcal{I}, r \in \Xi\} \tag{4}$$

While IA has been proven effective, it targets basic RecSys and may not be aligned with real-world scenarios. Thus, we formulate Comprehensive Attack as follows.

Definition 4 (**Comprehensive Attack (CA)**). *To align with real-world scenarios, we formulate Comprehensive Attack to focus on promoting the target item i_t to its target audience $\mathcal{U}_{TA}$ over competing products $\mathcal{I}_{compete}$ with the* Comprehensive Attack loss $\mathcal{L}_{CA}$ *defined as follows.*

$$\mathcal{L}_{CA} = \frac{1}{|\mathcal{U}_{TA}|} \sum_{u \in \mathcal{U}_{TA}} \sum_{i_c \in \mathcal{I}_{compete}} SELU(\mathcal{R}_{(u,i_c)} - \mathcal{R}_{(u,i_t)}), \tag{5}$$

where the scaled exponential linear units (SELU) [69] are used to emphasize the individual terms with the target item i_t losing to competing items, i.e., $\mathcal{R}_{(u,i_c)} - \mathcal{R}_{(u,i_t)} \geq 0$.

Furthermore, according to common practice, we formulate Comprehensive Attack to leverage a set of customer base *users $\mathcal{U}_{base} \subset \mathcal{U}$ besides the fake users $\mathcal{U}_{fake}$, and a set of* company products *items $\mathcal{I}_{product} \subset \mathcal{I}$ to facilitate the* comprehensive

poisoning actions on both the rating and the heterogeneous graph information.

Specifically, the capacity set $\mathcal{C}_{CA}$ consists of hiring users from the customer base $\mathcal{U}_{base}$ *to rate the target item with a preset rating $\hat{r} = 5$, hiring users from the* customer base *to connect to each fake account (on $\mathcal{G}_U$) and selecting items from the attacker's* company products $\mathcal{I}_{product}$ *to connect to the target item (on $\mathcal{G}_I$). The budget constraint restricts how many users (items) may be hired (selected).*

$$\begin{aligned}\mathcal{C}_{CA} = &\{\,(u, i_t, \hat{r}) \mid u \in \mathcal{U}_{base}\,\} \\ &\cup \{\,(u, u_f) \mid u \in \mathcal{U}_{base},\ u_f \in \mathcal{U}_{fake}\,\} \\ &\cup \{\,(i, i_t) \mid i \in \mathcal{I}_{product}\,\}\end{aligned} \tag{6}$$

For comparison, note that CA does not spam fake ratings with fake users like IA but explores vulnerabilities in the heterogeneous graph information.

Ultimately, we aim to investigate the scenario where an attacker faces one or many opponents. To account for the opponents' actions, we formulate Multiplayer Comprehensive Attack (MCA), where an attacker can optimize his poisoning plan by considering the expected (future) poisoning plans of his opponents. Notice that MCA inherits the same loss $\mathcal{L}_{CA}$ and the capacity (and budget constraint) $\mathcal{C}_{CA}$ from CA. However, while CA ignores other opponents, MCA anticipates the subsequent poisoning actions by opponents (who are conducting CAs) based on their respective objectives and capacities. Not solvable by the bi-level optimization (Definition 2), MCA is formulated by the following *multilevel optimization*.

Definition 5 (Multiplayer Comprehensive Attack (MCA)). *In general, an attacker may face N individual opponents, resulting in an $(N+1)$-player game, and thereby an $(N+2)$-level optimization problem. Given the attacker p and his opponents $q_i, i \in [1, N]$. Let $\mathcal{L}_{CA}^{p}$ and $\mathcal{L}_{CA}^{q_i}$ be the* Comprehensive Attack loss *and $\mathcal{C}_{CA}^{p}$ and $\mathcal{C}_{CA}^{q_i}$ be the capacity of the attacker (leader) p and the opponent (follower) q_i, respectively. The attacker p aims to solve:*

$$\begin{aligned}&\min_{\mathcal{X}^p \subseteq \mathcal{C}_{CA}^{p}} \mathcal{L}_{CA}^{p}(\mathcal{R}(\theta^*, \hat{\mathcal{G}})) \\ &\mathcal{X}^{q_1} \in \arg\min_{\mathcal{X} \subseteq \mathcal{C}_{CA}^{q_1}} \mathcal{L}_{CA}^{q_1}(\mathcal{R}(\theta^*, \hat{\mathcal{G}})) \\ &\qquad\vdots \\ &\mathcal{X}^{q_N} \in \arg\min_{\mathcal{X} \subseteq \mathcal{C}_{CA}^{q_N}} \mathcal{L}_{CA}^{q_N}(\mathcal{R}(\theta^*, \hat{\mathcal{G}})) \\ &given\ \theta^* \in \arg\min_{\theta} \mathcal{L}_{train}(\mathcal{R}(\theta, \hat{\mathcal{G}}), \hat{\mathbf{R}}) + \lambda\|\theta\|^2,\end{aligned} \tag{7}$$

where similar to (2) in Definition 2, $\mathcal{R}$ is the RecSys result parameterized by θ, $\hat{\mathbf{R}}$ and $\hat{\mathcal{G}}$ are the rating records and the heterogeneous graph poisoned by all $N+1$ players, while $\mathcal{X}^p$ and $\mathcal{X}^{q_i}$ are the sets of poisoning actions selected by p and q_i, respectively.

IV. The MSOPDS Method

Here, we introduce the *Multilevel Stackelberg Optimization over Progressive Differentiable Surrogate (MSOPDS)*, a framework that allows an attacker to plan his poisoning plan against subsequent opponents' poisoning plans to solve MCA. We first introduce the formulation of the *importance vector* to transform the original discrete set selection problem into continuous vector optimization. Then, we introduce the MSO framework which updates the attacker's poisoning plan while simulating the corresponding opponents' poisoning plans and estimating the total poisoning effect on the Het-RecSys, with all player's plans instantiated by *importance vector*s. We then present PDS, which facilitates computations of the required derivatives for the update rules calculated in MSO for each player's poisoning plan. Finally, we detail the algorithm for MSOPDS and present the pseudo-code in Algorithm 1.

A. The importance vector

To facilitate a gradient-based approach that iteratively adjusts intermediate plans by appropriate update rules, we formulate the *importance vector* $\mathbf{X} \in \mathbb{R}^{|\mathcal{C}|}$ where each element x indicates the *priority* of the corresponding candidate poisoning action in $\mathcal{C}$. Given the budget constraints for three types of poisoning actions, the importance vector $\mathbf{X}$ maps to a poisoning plan $\mathcal{X}$ consisting of the actions corresponding to the top-valued elements in $\mathbf{X}$. In the following, we elucidate how an attacker would optimize $\mathbf{X}$ with MSO and PDS.

B. Multilevel Stackelberg Optimization (MSO) framework

1) two-player game: We first explore the analytical solution to MCA in the simplified two-player setting between the attacker and a single opponent, then extend to the full solution of N opponents. Under the two-player setting, (7) becomes

$$\begin{aligned}&\min_{\mathcal{X}^p \subseteq \mathcal{C}^p} \mathcal{L}^p(\mathcal{R}(\theta^*, \hat{\mathcal{G}})) \\ &\mathcal{X}^q \in \arg\min_{\mathcal{X} \subseteq \mathcal{C}^q} \mathcal{L}^q(\mathcal{R}(\theta^*, \hat{\mathcal{G}})) \\ &given\ \theta^* \in \arg\min_{\theta} \mathcal{L}_{train}(\theta, \mathcal{R}(\theta^*, \hat{\mathcal{G}}), \hat{\mathbf{R}}) + \lambda\|\theta\|^2,\end{aligned} \tag{8}$$

where subscripts $_{CA}$ are removed from $\mathcal{C}$ and $\mathcal{L}$ for simplicity.

As displayed in (8), the attacker optimizes his poisoning plan $\mathcal{X}^p$ (top level of (8)) *over* a bi-level optimization process of his opponent q solving for $\mathcal{X}^q$ (middle and lower level of (8)). Accordingly, if the attacker decides on a specific poisoning plan $\mathcal{X}^p$, he may simulate the final RecSys result $\mathcal{R}$ by solving his opponent's poisoning plan with the rating records $\mathbf{R}$ and heterogeneous graph $\mathcal{G}$ already poisoned with $\mathcal{X}^p$. Following prior works [3] and given the poisoning plans $\mathcal{X}^p$ and $\mathcal{X}^q$ decided by the importance vectors $\mathbf{X}^p$ and $\mathbf{X}^q$, we present the basic update rule for the opponent as follows.

$$\mathbf{X}^q \leftarrow \mathbf{X}^q - \eta^q \frac{\partial \mathcal{L}^q}{\partial \mathbf{X}^q}, \tag{9}$$

where η^q is the step size of the opponent and the partial derivative $\partial\mathcal{L}^q/\partial\mathbf{X}^q$ represents how a change in $\mathbf{X}^q$ directly affects $\mathcal{L}^q$.

After solving for $\mathbf{X}^q$, an idea is to update $\mathbf{X}^p$ with update rules similar to (9), given $\mathbf{R}$ and $\mathcal{G}$ poisoned by $\mathcal{X}^q$. However, such an approach fail to consider that the current $\mathbf{X}^q$ is solved based on a given $\mathbf{X}^p$, and would change again after $\mathbf{X}^p$ is

updated. Intuitively, such an approach may not lead to an equilibrium but rather a never-ending cycle where one player constantly tries to outmaneuver the other.

We thus present the following update rule for the attacker.

$$\mathbf{X}^p \leftarrow \mathbf{X}^p - \eta^p \frac{d\mathcal{L}^p}{d\mathbf{X}^p}, \tag{10}$$

where

$$\frac{d\mathcal{L}^p}{d\mathbf{X}^p} = \frac{\partial \mathcal{L}^p}{\partial \mathbf{X}^p} - \frac{\partial \mathcal{L}^p}{\partial \mathbf{X}^q}\frac{\partial \mathbf{X}^q}{\partial \mathbf{X}^p} \tag{11}$$

is the *total derivative* of $\mathcal{L}^p$ *w.r.t.* $\mathbf{X}^p$, representing how the attacker's actions $\mathbf{X}^p$ influence his objective $\mathcal{L}^p$ directly in the first partial derivative term, and indirectly due to the subsequent opponent reactions in the second term. By utilizing the *total derivative*, the attacker is able to incorporate and adapt to the anticipated opponent's actions. However, as the term $\frac{\partial \mathbf{X}^q}{\partial \mathbf{X}^p}$ cannot be easily computed, we aim to expand (11) into a formula that is *computable*.

We first consider a temporary construction where after each attacker update step, the opponent updates by (9) until convergence, i.e., $\partial \mathcal{L}^q / \partial \mathbf{X}^q = 0$. As shown in left of Fig. 3, such an optimality condition implicitly defines $\mathbf{X}^q$ as a function of $\mathbf{X}^p$ through the dependence of $\mathcal{L}^q$ on $\mathbf{X}^p$ by the implicit function theorem [70]. We find

$$\frac{d}{d\mathbf{X}^p}\left(\frac{\partial \mathcal{L}^q}{\partial \mathbf{X}^q}\right) = \frac{\partial^2 \mathcal{L}^q}{\partial \mathbf{X}^{q\,2}}\frac{\partial \mathbf{X}^q}{\partial \mathbf{X}^p} + \frac{\partial^2 \mathcal{L}^q}{\partial \mathbf{X}^p \partial \mathbf{X}^q} = 0 \tag{12}$$

and thus

$$\frac{d\mathcal{L}^p}{d\mathbf{X}^p} = \frac{\partial \mathcal{L}^p}{\partial \mathbf{X}^p} - \frac{\partial \mathcal{L}^p}{\partial \mathbf{X}^q}\left(\frac{\partial^2 \mathcal{L}^q}{\partial \mathbf{X}^{q\,2}}\right)^{-1}\frac{\partial^2 \mathcal{L}^q}{\partial \mathbf{X}^p \partial \mathbf{X}^q} \tag{13}$$

by rearranging the terms in (12) and applying the resulting formulation for $\partial \mathbf{X}^q / \partial \mathbf{X}^p$ to (10).

Equation (13) is attained under the strict assumption that the opponent's plan $\mathbf{X}^q$ is updated to convergence after every attacker update. Nevertheless, such a requirement inevitably causes the entire algorithm to be very slow. Thus, based on recent results reported by Fiez et al. [71], we proceed to relax the requirement and construct the procedure where both $\mathbf{X}^p$ and $\mathbf{X}^q$ are *updated simultaneously* by (13) and (9), respectively, on the condition that $\eta^p < \eta^q$. With the formal theoretical analysis detailed in Section V, a high-level explanation is presented as follows. Under the former strict requirement, the opponent (i.e., the follower) always responds with a local best strategy at each iteration, forming an *optimal trajectory*. As illustrated in the right of Fig. 3, in the relaxed case, (9) still "pulls" $\mathbf{X}^q$ towards the optimal trajectory, while the attacker update changes the loss landscape of $\mathcal{L}^q$ and "pushes" $\mathbf{X}^q$ away from it. Intuitively, $\eta^p < \eta^q$ guarantees that the former overcomes the latter such that the new trajectory of $\mathbf{X}^q$ asymptotically converges to the *optimal trajectory*.

2) General multiplayer game: To efficiently solve the general multiplayer MCA game with N opponents, we focus on the interaction between the attacker and each opponent. In particular, since each opponent conducts Comprehensive Attack and is oblivious to the other players, we may reapply

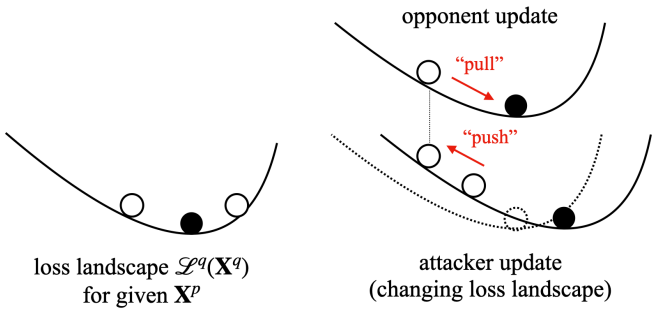


Fig. 3: Left: given the attacker's poison $\mathbf{X}^p$, the opponent's loss landscape $\mathcal{L}^q(\mathbf{X}^q)$ is fixed. Intuitively, there is a corresponding optimal poison plan by the opponent (black circle) but arbitrarily many suboptimal points (white circles). Right: An illustration of "push" vs. "pull". The opponent's update step takes it closer to the optimal, resembling a "pull" to the optimal position. However, the attacker's update step may shift the loss landscape such that the corresponding opponent's position is further away from optimal, resembling a "push".

the attacker update rule (13) with minor adjustments. Following [72], we simplify (7) by flattening the lower-level objectives of opponents $q_1 \cdots q_N$ into a single-level optimization. With $\mathbf{X}$s denoting the importance vectors as above, we can rewrite (13) into a multi-opponent version as follows.

$$\frac{d\mathcal{L}^p}{d\mathbf{X}^p} = \frac{\partial \mathcal{L}^p}{\partial \mathbf{X}^p} - \sum_{i=1}^{N} \frac{\partial \mathcal{L}^p}{\partial \mathbf{X}^{q_i}}\left(\frac{\partial^2 \mathcal{L}^{q_i}}{\partial \mathbf{X}^{q_i\,2}}\right)^{-1}\frac{\partial^2 \mathcal{L}^{q_i}}{\partial \mathbf{X}^p \partial \mathbf{X}^{q_i}}, \tag{14}$$

where the second term in (13) is extended into the summation that covers the indirect effects of all opponents. Thus, MSO solves (7) in the general setting by iteratively updating $\mathbf{X}^p$ by (10) with the *total derivative* $\frac{d\mathcal{L}^p}{d\mathbf{X}^p}$ defined by (14) and while each simulated opponent poisoning plan $\mathbf{X}^{q_i}$ is still updated by $\frac{\partial \mathcal{L}^{q_i}}{\partial \mathbf{X}^{q_i}}$ as in (9).

C. Progressive Differentiable Surrogate

To compute the required derivatives in the update rules (13) and (9), we introduce the Progressive Differentiable Surrogate (PDS). Following the design of GNN-based RecSys design [12], [68], PDS iteratively computes the user $\boldsymbol{h}_u$ and item $\boldsymbol{h}_i$ embeddings to obtain the final user $\boldsymbol{h}_u^f$ and item $\boldsymbol{h}_u^f$ embeddings, through graph convolution on the respective graphs $\mathcal{G}_U$ and $\mathcal{G}_I$. Accordingly, the RecSys ratings prediction $\mathcal{R}_{(u,i)} \equiv \boldsymbol{h}_u^f \cdot \boldsymbol{h}_i^f$ is obtained through dot products of between those final embeddings.

Intuitively, we must *incorporate* the importance vectors $\mathbf{X}$ throughout the training process of PDS for $\mathbf{X}$ to influences the RecSys results and eventually the loss objectives. As illustrated in Fig. 4, PDS first obtains the *binarized* importance vector $\hat{\mathbf{X}}$ (representing a tentative attack plan $\mathcal{X}$) by setting the elements of $\mathbf{X}$ with the largest values to 1 and others to 0, according

to budget. Then, PDS utilizes $\hat{\mathbf{X}}$ in the graph convolutional processes for the final embeddings as follows.

$$\begin{aligned} \boldsymbol{h}_u^f &= \mathbf{W}_U^\top \left(\boldsymbol{h}_u \oplus \sum_{n \in \mathcal{N}_U(u)} \mathbb{1}_{\mathcal{C}} \hat{x}_U(u,n) \frac{\boldsymbol{h}_n}{|\mathcal{N}_U(u)|} \right) \\ \boldsymbol{h}_i^f &= \mathbf{W}_I^\top \left(\boldsymbol{h}_i \oplus \sum_{m \in \mathcal{N}_I(i)} \mathbb{1}_{\mathcal{C}} \hat{x}_I(i,m) \frac{\boldsymbol{h}_m}{|\mathcal{N}_I(i)|} \right), \end{aligned} \tag{15}$$

where $\mathbf{W}_U$ and $\mathbf{W}_I$ are trainable projection matrices, $\oplus$ denotes concatenation, $\mathcal{N}_U(v)$ and $\mathcal{N}_I(i)$ represent the first-hop neighbors of u and i in $\mathcal{G}_U$ and $\mathcal{G}_I$, respectively. $\hat{x}_U(u,n)$ and $\hat{x}_I(i,m)$ are element values of $\hat{\mathbf{X}}$ that corresponds to the poisoning edge between u and n in the social network and between i and m in the item graph, respectively, whereas $\mathbb{1}_{\mathcal{C}}$ is a selection function that defaults to one, but adopts element value of the binarized importance vector ($\hat{x}_U(u,n)$ or $\hat{x}_I(i,m)$) if the link ((u,n) or (i,m) is in the candidate set $\mathcal{C}$. In other words, the selection function applies the selection decision presented in the binarized element values into the graph convolution process and corresponds to whether the surrogate model perceives that the corresponding edge exists.

Furthermore, PDS replaces $\mathcal{L}_{train}$ in (1) with a $\hat{\mathbf{X}}$-modulated loss $\mathcal{L}_{train-PDS}$ as follows.

$$\begin{aligned} \mathcal{L}_{train-PDS} = &\sum_{(u,i,r) \in \mathbf{R}} \left(\boldsymbol{h}_u^f \cdot \boldsymbol{h}_i^f - r \right)^2 \\ &+ \sum_{(u,i) \in \mathcal{C}_R} \hat{x}_R(u,i) \left(\boldsymbol{h}_u^f \cdot \boldsymbol{h}_i^f - \hat{r} \right)^2, \end{aligned} \tag{16}$$

where $\mathbf{R}$ is the real ratings records, $\mathcal{C}_R$ is the set of candidate poisoning actions that include adding item ratings (with real or fake users), $\hat{x}_R(u,i)$ are element values of $\hat{\mathbf{X}}$ that correspond to the poisoning action of adding a rating $r_{u,i}$ with user u on item i, and $\hat{r}$ is a predefined rating value.[4]

With (15) and (16), the binarized importance vector $\hat{\mathbf{X}}$ corresponding to poisoning actions on the social network $\mathcal{G}_U$ and item graph $\mathcal{G}_I$ affects the final user and item embeddings through (15), while those corresponding to poisoning actions on the rating records $\mathbf{R}$ affects the training process through the training loss (16). Since *all* elements of $\mathbf{X}$ are utilized in either (15) or (16), no matter the element value being 0 or 1, derivatives with respect to the whole vector may be computed.

D. The complete MSOPDS framework

Combining MSO and PDS, we present the complete MSOPDS framework. Fig. 5 presents an illustrative example of one iteration of MSOPDS for the two-player case. As shown in Fig. 5, the importance vectors $\mathbf{X}^p$ and $\mathbf{X}^q$ are binarized into $\hat{\mathbf{X}}^p$ and $\hat{\mathbf{X}}^p$, then fed into the PDS to simulate the poisoned RecSys result and to evaluate the poisoning objectives of the attacker $\mathcal{L}^p$ and the opponent $\mathcal{L}^q$. Finally, by the update rules of MSO, we update $\mathbf{X}^p$ by computing the *total derivative* of $\mathcal{L}^p$ w.r.t. $\hat{\mathbf{X}}^p$ and the *partial derivative* of $\mathcal{L}^q$ w.r.t. $\hat{\mathbf{X}}^q$.

[4]Note that $\hat{r}$ can be set to a high value to influence PDS to predict higher ratings (i.e., promotion), or vice versa.

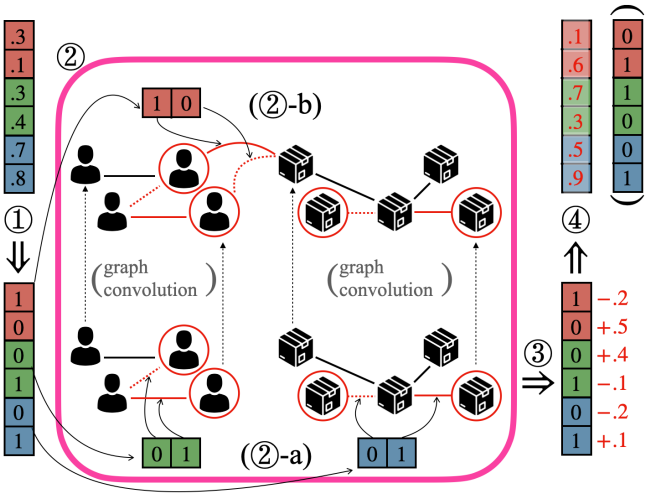


Fig. 4: Illustrative example of PDS. From the top left: ① the importance vector is binarized to select 1 of each type of poisoning action (budget constraint). ② the binary result is incorporated into the RecSys training process: (②-a) poison edges regulate the graph convolution on social network and item graph; (②-b) poison ratings modulate the training loss; ③ backpropagation through the recorded training process (in ②) computes the required derivatives according to the update rule (9) or (10);④ finally, the importance vectors are adjusted by the update rules. Comparing the binarized results before (bottom left) and after (top right) this iteration, the poison action is changed accordingly.

Finally, Algorithm 1 presents the pseudo-code for the two-player case. As presented in Algorithm 1, an inner loop of PDS training is embedded within an outer loop of K iterations of updates for the importance vectors $\mathbf{X}^p$ and $\mathbf{X}^q$ following the MSO update rules. Notice that in step 2, *all* candidate poisoning is inserted into the fully poisoned rating records $\mathbf{R}'$ and heterogeneous graph $\mathcal{G}'$, which is later regulated by the element values of the binarized importance vectors during the PDS training process. Therefore, by storing the inner loop RecSys training computations (step 6), first-order derivatives of the losses w.r.t. the binarized importance vectors ($\frac{\partial \mathcal{L}^p}{\partial \hat{\mathbf{X}}^p}$, $\frac{\partial \mathcal{L}^p}{\partial \hat{\mathbf{X}}^q}$, and $\frac{\partial \mathcal{L}^q}{\partial \hat{\mathbf{X}}^q}$) can be obtained via backpropagation through the stored computation graph in step 8.

To reduce the computation cost of deriving the full Jacobian matrices, we defer the realization of the Jacobian matrices and integrate the calculation of vector-Jacobian products into each iteration when solving $\boldsymbol{\xi}$ for the linear equation $\boldsymbol{\xi} \frac{\partial^2 \mathcal{L}^q}{\partial \mathbf{X}^{q\,2}} = \frac{\partial \mathcal{L}^p}{\partial \mathbf{X}^q}$ by the conjugate gradient method [73] (step 9).

Finally, following the update rules ((10) for the attacker and (9) for the opponent) as well as the calculations of (13) in MSO, step 10 and 11 updates the attacker's importance vector $\mathbf{X}^p$ and the opponent's importance vector $\mathbf{X}^q$, respectively.

Remark. To perform MSOPDS for multiple opponents, we modify Algorithm 1 according to (14). Besides applying the

Algorithm 1: MSOPDS under the two player setting.

Input: Rating records $\mathbf{R}$, graph records $\mathcal{G}$; Comprehensive Attack losses $\mathcal{L}^p$ and $\mathcal{L}^q$; capacity sets $\mathcal{C}^p$ and $\mathcal{C}^q$ (with budget constraints represented as $\mathcal{B}^p$ and $\mathcal{B}^q$); step sizes η^p and η^q; outer and the inner loop max iteration K and L; PDS parameter θ.

Assert: $0 < \eta^p < \eta^q$

Output: Attacker's action plan $\mathcal{X}^p$.

1: Create importance vectors $\mathbf{X}^p$ and $\mathbf{X}^q$ from $\mathcal{C}^p$ and $\mathcal{C}^q$.
2: Create $\mathbf{R}'$ and $\mathcal{G}'$ by inserting all poisoning actions in $\mathcal{C}^p$ and $\mathcal{C}^q$ into the rating record $\mathbf{R}$ and the heterogeneous graph $\mathcal{G}$;
3: **for** $k = 1$ **to** K **do**
4: Binarize $\mathbf{X}^p$ and $\mathbf{X}^q$ into $\hat{\mathbf{X}}^p$ and $\hat{\mathbf{X}}^q$ by $\mathcal{B}^p$ and $\mathcal{B}^q$.
5: **for** $l = 1$ **to** L **do**
6: Update $\theta^{(l)}$ by (1) with the training loss (16) and graph convolution (15) modulated by $\hat{\mathbf{X}}^p$ and $\hat{\mathbf{X}}^q$.
`// store the computation process.`
7: Derive the Comprehensive Attack losses $\mathcal{L}^p$ and $\mathcal{L}^q$ for both players with the trained (poisoned) RecSys results $\mathcal{R}(\theta^{(L)}, \mathcal{G}'')$.
8: Compute first-order partial derivatives $\frac{\partial \mathcal{L}^p}{\partial \hat{\mathbf{X}}^p}$, $\frac{\partial \mathcal{L}^p}{\partial \hat{\mathbf{X}}^q}$, and $\frac{\partial \mathcal{L}^q}{\partial \hat{\mathbf{X}}^q}$ by backpropagation through the stored process.
`// retain computation graph for second-order vector-Jacobian product.`
9: Solve $\boldsymbol{\xi}$ for $\boldsymbol{\xi} \frac{\partial^2 \mathcal{L}^q}{\partial \hat{\mathbf{X}}^{q\,2}} = \frac{\partial \mathcal{L}^p}{\partial \hat{\mathbf{X}}^q}$.
`// Calculate vector-Jacobian product within the conjugate gradient method.`
10: Update attacker $\mathbf{X}^p \leftarrow \mathbf{X}^p - \eta^p \left(\partial \mathcal{L}^p / \partial \hat{\mathbf{X}}^p - \boldsymbol{\xi} \frac{\partial^2 \mathcal{L}^q}{\partial \hat{\mathbf{X}}^p \partial \hat{\mathbf{X}}^q} \right)$.
`// Calculate vector-Jacobian product.`
11: Update opponent $\mathbf{X}^q \leftarrow \mathbf{X}^q - \eta^q \partial \mathcal{L}^q / \partial \hat{\mathbf{X}}^q$.

same η^q for all opponents, the input as well as steps 1, 2, 4, 6 and 7 are straightforwardly adjusted to *include all opponent* q_i. Step 8 includes all first-order partial derivative described in (14) for the respective $\hat{\mathbf{X}}$s. Step 9 and 11 are repeated for each q_i while step 10 is adjusted according to (14).

Besides, we also present an ablation method named *Bi-level Optimization over Progressive Differentiable Surrogate (BOPDS)* by combining the bi-level optimization (Definition 2) with PDS to conduct Comprehensive Attack, later used as the opponents' method in Section VI. We modify Algorithm 1 to perform the single-player BOPDS by removing the opponent q from the input as well as steps 1, 2, 4, 6 and 7, replacing the final steps $8-11$ by only computing $\frac{\partial \mathcal{L}^p}{\partial \hat{\mathbf{X}}^p}$ and updating the attacker's importance vector by $\mathbf{X}^p \leftarrow \mathbf{X}^p - \eta^p \frac{\partial \mathcal{L}^p}{\partial \hat{\mathbf{X}}^p}$.

V. Theoretical Analysis

We first provide the time complexity of MSOPDS.

Theorem 1. *With MSOPDS, the time complexity are reduced from* $\Omega(|\theta|^{|\mathbf{X}^p||\mathbf{X}^q|})$ *to* $O(|\theta| + |\mathbf{X}^p||\mathbf{X}^q|)$*, where* $|\cdot|$ *represents the dimension of the model parameters or importance vectors.*

Proof. Since we avoid enumerating all possible combinations, the computational complexities of MSOPDS are reduced from the brute-force method. In particular, $O(|\theta|)$ is the complexity of calculating $\frac{\partial \mathcal{L}}{\partial X}$ with the reverse-mode algorithmic differentiation, which is proportional to the surrogate model size $|\theta|$ [74]. $O(|\mathbf{X}^p||\mathbf{X}^q|)$ is caused by the conjugate gradient method, which is proportional to the size of the Jacobian matrix $\frac{\partial^2 \mathcal{L}^q}{\partial \mathbf{X}^p \partial \mathbf{X}^q}$ [73]. □

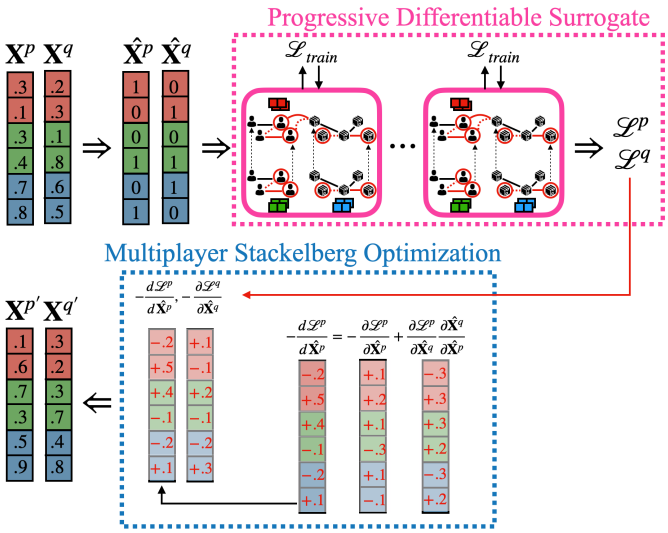


Fig. 5: Illustration of an iteration in MSOPDS. We highlight the two components of MSOPDS, namely, Multilevel Stackelberg Optimization (MSO) and Progressive Differentiable Surrogate (PDS), by blue and pink box, respectively.

Next, we prove the existence of Stackelberg game equilibrium and the convergence to such equilibrium in terms of player strategy optimization. For simplicity, we focus on the two-player Stackelberg game between an attacker p (leader) and one opponent q (follower).

Definition 6 (**Stackelberg Equilibrium** [75])**.** *Let* $\mathcal{C}^p$ *and* $\mathcal{C}^q$ *be the capacity of the leader and the follower, respectively. The poisoning plan* $\mathbf{X}^{p*} \in \mathcal{C}^p$ *is a Stackelberg solution for the leader if* $\forall \mathbf{X}^p \in \mathcal{C}^p$*,*

$$\inf_{\mathbf{X}^q \in R_{\mathcal{C}^q}(\mathbf{X}^{p*})} \mathcal{L}^p(\mathbf{X}^{p*}, \mathbf{X}^q) \geq \inf_{\mathbf{X}^q \in R_{\mathcal{C}^q}(\mathbf{X}^p)} \mathcal{L}^p(\mathbf{X}^p, \mathbf{X}^q), \tag{17}$$

where $R_{\mathcal{C}^q}(\mathbf{X}^p) = \arg\max_{Y \in \mathcal{C}^q} \mathcal{L}^q(\mathbf{X}^p, Y)$ *is the optimal solution set for the follower. More precisely,*

$$(\mathbf{X}^{p*}, \mathbf{X}^{q*}), \forall \mathbf{X}^{q*} \in R_{\mathcal{C}^q}(\mathbf{X}^{p*}) \tag{18}$$

are the Stackelberg equilibriums.

Following [71], we utilize the differential Stackelberg equilibrium as follows.

Definition 7 (**Differential Stackelberg Equilibrium** [71])**.** *A pair of leader and follower poisons* $(\mathbf{X}^{p*}, \mathbf{X}^{q*}) \in (\mathcal{C}^p, \mathcal{C}^q)$ *is a Differential Stackelberg Equilibrium if*

$$\frac{d^2}{d\mathbf{X}^{p\,2}} \mathcal{L}^p(\mathbf{X}^{p*}, \mathbf{X}^{q*}) > 0, \quad \frac{\partial^2}{\partial \mathbf{X}^{q\,2}} \mathcal{L}^q(\mathbf{X}^{p*}, \mathbf{X}^{q*}) > 0, \tag{19}$$

$$\frac{d}{d\mathbf{X}^p} \mathcal{L}^p(\mathbf{X}^{p*}, \mathbf{X}^{q*}) = \frac{\partial}{\partial \mathbf{X}^q} \mathcal{L}^q(\mathbf{X}^{p*}, \mathbf{X}^{q*}) = 0. \tag{20}$$

For reasonably smooth loss functions $\mathcal{L}^p, \mathcal{L}^q$, (19) provides the necessary conditions for the implicit mapping utilized in (13), while the partial differentiation in (20) recreates the

condition $R_{\mathcal{C}^q}$ in Definition 6, where the total differentiation corresponds to the optimality of X^p in (17).

Intuitively, with Definition 7, we can approach Stackelberg equilibrium by updating the leader's action to minimize the total differentiation and the follower's action to minimize the partial differentiation. This corresponds to the calculations proposed for MSO in Section IV-B.

Theorem 2. *Stackelberg equilibrium exists for MCA between one attacker (p) and one opponent (q).*

Proof. In MCA, a GNN-based Het-RecSys is trained (poisoned) by $\mathbf{X}^p$ and $\mathbf{X}^q$. In particular, the training loss is an evaluation of how close the predicted result of the Het-RecSys $\mathcal{R}$ follows the given rating records $\mathbf{R}$. The training process of Het-RecSys aims to minimize $\mathcal{L}_{train}$ to an optimal value. According to Theorem 1 in [76], the convergence rate of GNN training with error rate ϵ is $O(\frac{1}{\epsilon^2}\log(\frac{1}{\epsilon}))$ by the Polyak-Lojasiewicz inequality [77], namely that the squared norm of the gradient $\nabla\mathcal{L}_{train}$ lower bounds the loss value at any iteration $|\mathcal{L}_{train,t} - \mathcal{L}_{train,t-1}|$. Since $\mathcal{L}_{train}$ regulates the Het-RecSys prediction results $\mathcal{R}$, it also determines the poisoning objectives $\mathcal{L}^p$ and $\mathcal{L}^q$. Therefore, a set of $\mathbf{X}^p$ and $\mathbf{X}^q$ deterministically leads to a set of poisoning results $\mathcal{L}^p$ and $\mathcal{L}^q$ due to the convergence of Het-RecSys. Since the candidate sets $\mathcal{C}^p$ and $\mathcal{C}^q$ are finite sets and every possible strategy pair $(\mathbf{X}^p, \mathbf{X}^q)$ is finite under a given budget, an optimal strategy, i.e., Stackelberg equilibrium, exists over the finite set [78]. □

We continue to assert the convergence of MSOPDS towards differential Stackelberg equilibrium of MCA.

Theorem 3. *MSOPDS converges to a differential Stackelberg equilibrium in MCA.*

Proof. Consider a differential Stackelberg equilibrium $\mathbf{X}^* \equiv (\mathbf{X}^{p*}, \mathbf{X}^{q*})$ such that $\frac{d\mathcal{L}^p(\mathbf{X}^*)}{d\mathbf{X}^p} = \frac{\partial\mathcal{L}^q(\mathbf{X}^*)}{\partial\mathbf{X}^q} = 0$, $\text{spec}(\frac{d^2\mathcal{L}^p(\mathbf{X}^*)}{d{\mathbf{X}^p}^2}) \subset \mathbb{R}^\circ_+$ and $\text{spec}(\frac{\partial^2\mathcal{L}^q(\mathbf{X}^*)}{\partial{\mathbf{X}^q}^2}) \subset \mathbb{R}^\circ_+$. Since $\det(\frac{\partial^2\mathcal{L}^q(\mathbf{X}^*)}{\partial{\mathbf{X}^q}^2}) \neq 0$ and $\frac{\partial}{\partial\mathbf{X}^q}\mathcal{L}^q(\mathbf{X}^{p*}, \mathbf{X}^{q*}) = 0$, by the implicit function theorem [70], there exists a neighborhood $\mathcal{N}_1$ of $\mathbf{X}^p$ and a unique function $f : \mathcal{N}_1 \to \mathbb{R}^{|\mathcal{C}^q|}$ such that $\frac{\partial}{\partial\mathbf{X}^q}\mathcal{L}^q(\mathbf{X}^p, f(\mathbf{X}^p)) = 0\ \forall\mathbf{X}^p \in \mathbb{R}^{|\mathcal{C}^p|}$ and $f(\mathbf{X}^{p*})) = \mathbf{X}^{q*}$.[5]

Due to the fact that eigenvalues vary continuously, there exists a neighborhood $\mathcal{N}_2$ of $\mathbf{X}^p$ where $\frac{d^2}{d{\mathbf{X}^p}^2}\mathcal{L}^p(\mathbf{X}^{p*}, f(\mathbf{X}^{p*})) > 0$ and $\frac{\partial^2}{\partial{\mathbf{X}^q}^2}\mathcal{L}^q(\mathbf{X}^{p*}, f(\mathbf{X}^{p*})) > 0$. Let $\mathcal{U}_1 \subseteq \mathcal{N}_1 \cap \mathcal{N}_2$ be the non-empty, open set whose closure is contained in $\mathcal{N}_1 \cap \mathcal{N}_2$. Since $\frac{\partial^2}{\partial{\mathbf{X}^q}^2}\mathcal{L}^q(\mathbf{X}^{p*}, \mathbf{X}^{q*}) > 0$, there exists an open neighborhood $\mathcal{U}_2$ of $\mathbf{X}^{q*}$ such that $\frac{\partial^2}{\partial{\mathbf{X}^q}^2}\mathcal{L}^q(\mathbf{X}^p, \mathbf{X}^q) > 0\ \forall(\mathbf{X}^p, \mathbf{X}^q) \in \mathcal{U}_1 \times \mathcal{U}_2$. Since we have set the step-size of the attacker η^p and the subsequent opponent η^q to be $\eta^p < \eta^q$ in MSOPDS, according to (Lemma G.2 of [71]), the optimization process of MSOPDS on $(\mathbf{X}^p, \mathbf{X}^q)$ converges to $(\mathbf{X}^{p*}, \mathbf{X}^{q*})$. □

[5]Empirically, the numerical values of the total derivative $\frac{d\mathcal{L}^p}{d\mathbf{X}^p}$ and the partial derivative $\frac{\partial\mathcal{L}^q}{\partial\mathbf{X}^q}$ are all within reasonable range throughout the MSOPDS iterations, implying that they are L_1 and L_2-Lipschitz, respectively.

In Theorem 3 above, we prove that MSOPDS converges to an equilibrium in MCA. However, the equilibrium may not be unique since there may exist multiple poisoning strategies pairs for $\mathbf{X}^p$ and $\mathbf{X}^q$ that leads to the same result.

VI. Experiments

A. Experiment settings

1) Datasets and Threat Het-RecSys: We evaluate three real-world datasets, including **Ciao** [79] ($2,611$ users, $3,823$ items, $44,453$ ratings, and $49,953$ links), **Epinion** [80] ($1,929$ users, $9,962$ items, $12,612$ ratings, and $41,270$ links), and **LibraryThing** [81] ($1,108$ users, $8,583$ items, $19,615$ ratings, and 14508 links).[6] The links denote the number of edges in the social network $\mathcal{G}_U$. Following [12], the item graph $\mathcal{G}_I$ is created by connecting items that share over 50% of users that rated them in the rating record.

We experiment on a representative Het-RecSys, **ConsisRec** [12], which learns an embedding vector for each user and item node. To predict the rating of a user u on an item i, ConsisRec selectively aggregates the neighbor nodes on the heterogeneous graphs, i.e., the social network $\mathcal{G}_U$ and item graph $\mathcal{G}_I$, with an attention mechanism. Graph convolution operations are then performed over the aggregated nodes to obtain the final user and item embeddings. It then utilizes the dot product of the final embeddings for prediction and employs the Mean Square Error as the training loss in (1).

2) Demographic settings: We randomly select 5% of the user set $\mathcal{U}$ as the target audience $\mathcal{U}_{TA}$ and randomly sample 100 users from the user set $\mathcal{I}$ as the *customer base* $\mathcal{U}_{base}$ for each attacker and his opponents. We randomly select 50 items as the *competing items* $\mathcal{I}_{compete}$ and extract the item with the lowest average rating as the attacker's target item. Lastly, we randomly sample 100 items from non-competing items $\mathcal{I} \setminus \mathcal{I}_{compete}$ as the *company products* $\mathcal{I}_{product}$ for each attacker and his opponents.

3) Attacker's capability: We utilize a common budget parameter b, defaulting to $b = 5$, to control attacker's capability under IA and MCA. In both scenarios, the attacker creates fake users that amount to $b‰$ of the user set $|\mathcal{U}|$, with each fake user giving a top rating (5-star) [82] to promote the target items i_t. In addition to the fake user ratings, each attack scenario employs a different set of poisoning actions as follows.

- Under **Injection Attack (IA)**, with capacity $\mathcal{C}_{IA}$, the attacker rates 100 filler items with each fake user.
- Under **Multiplayer Comprehensive Attack (MCA)**, with capacity $\mathcal{C}_{CA}$, the attacker hires N real users from his *customer base* $\mathcal{U}_{base}$ to give top ratings (5-star) to i_t, connects each fake account to N real users of his *customer base* and selects N items of his *company products* $\mathcal{I}_{product}$ to connect to i_t on the item graph $\mathcal{G}_I$, where $N = b \times 5‰|\mathcal{U}|$.

Compared with IA, MCA reduce the number of fake ratings and replace them with new social and item links.

[6]We employ the standard preprocessing step to include only the users with at least 15 friends and at least 1 item rating.

TABLE III: Attacker's target item average predicted rating ($\bar{r}$) and Hit Rate@3 (HR@3) on ConsisRec facing a single opponent.

Dataset	Type	Method	$b=2$		$b=3$		$b=4$		$b=5$	
			$\bar{r}$	HR@3	$\bar{r}$	HR@3	$\bar{r}$	HR@3	$\bar{r}$	HR@3
Ciao	IA	None	2.1282	0.0154	2.1282	0.0154	2.1282	0.0154	2.1282	0.0154
		Random	2.6881	0.0692	2.6554	0.0538	2.8411	0.1154	2.8222	0.1000
		Popular	2.5508	0.0769	2.6948	0.0923	2.9072	0.1231	2.9403	0.1154
		PGA	2.5395	0.0615	2.5424	0.0538	2.7103	0.1000	2.8631	0.1154
		S-attack	2.6180	0.0846	2.8557	0.1231	2.7874	0.0923	2.9313	0.1154
		RevAdv	2.5250	0.0692	2.7005	0.0923	2.6618	0.0769	2.9862	0.1231
		Trial	2.6576	0.0769	2.5311	0.0538	2.8694	0.1000	3.0627	0.1462
	MCA	MSODS	**3.2100**	**0.1615**	**3.2177**	**0.1846**	**3.2022**	**0.1692**	**3.3953**	**0.2154**
Epinions	IA	None	1.2938	0.0000	1.2938	0.0000	1.2938	0.0000	1.2938	0.0000
		Random	1.9492	0.0000	2.6070	0.0104	2.4007	0.0104	2.5166	0.0208
		Popular	1.7655	0.0000	2.2960	0.0000	2.5757	0.0208	2.6094	0.0312
		PGA	1.9493	0.0000	2.3111	0.0000	2.3387	0.0000	2.4283	0.0000
		S-attack	1.8089	0.0000	2.2274	0.0000	2.3190	0.0208	2.5093	0.0312
		RevAdv	1.7721	0.0000	2.3980	0.0104	2.4116	0.0000	2.3946	0.0000
		Trial	1.8090	0.0000	2.3227	0.0000	2.3234	0.0000	2.4525	0.0104
	MCA	MSODS	**2.9252**	**0.1875**	**3.1662**	**0.2083**	**3.3865**	**0.2292**	**3.2414**	**0.1979**
LibraryThing	IA	None	1.7001	0.0000	1.7001	0.0000	1.7001	0.0000	1.7001	0.0000
		Random	1.9865	0.0213	2.0482	0.0106	2.1590	0.0106	2.4537	0.0638
		Popular	2.0421	0.0213	2.1894	0.0426	2.2663	0.0426	2.3879	0.0426
		PGA	2.0316	0.0213	2.1229	0.0319	2.3192	0.0426	2.3629	0.0426
		S-attack	2.0486	0.0213	2.2473	0.0638	2.3050	0.0426	2.3725	0.0319
		RevAdv	1.9276	0.0213	2.0768	0.0213	2.3096	0.0319	2.4007	0.0532
		Trial	1.9687	0.0213	2.0220	0.0319	2.3881	0.0532	2.2885	0.0426
	MCA	MSODS	**2.7410**	**0.2234**	**3.0174**	**0.3085**	**3.0541**	**0.3830**	**3.0420**	**0.3085**

4) Opponent goal and capability: We assume the opponents conduct a simplified Comprehensive Attack with budget $b_{op} = 2$.[7] In particular, each opponent selects real users from his customer base by BOPDS give 1-star ratings to the attacker's target item i_t, such that the target item is demoted among the competing items $\mathcal{I}_{compete}$.

5) Compared baseline methods: We compare the proposed method with several baselines with the parameters set according to the original papers. Note that **MSOPDS** is utilized under MCA, which anticipates subsequent attacks by opponents. On the other hand, the compared baseline attacks are all operated under the IA setting.[8]

- **None**: The attacker conducts no attack.
- **Random**: The attacker selects proxy items randomly.
- **Popular** [49], [84]: The attacker selects 90% random and 10% popular items as the proxy items.
- **Projected gradient ascent attack (PGA)** [13]: The attacker optimizes the poisoning attack over matrix factorization-based recommendation models.
- ***S*-attack** [52]: *S*-attack formulates the selection of proxy items as an optimization problem and targets on a graph-based recommender model. Each proxy item is rated with a normal distribution that matches the real user ratings.
- **Revisit Attack (RevAdv)** [3]: The state-of-the-art bilevel optimization approach with the gradient calculation conducted over the process of RecSys training over the generated poison data.
- **Trial Attack** [54]: The attack utilizes triple adversarial learning by extending the traditional GAN [85] into a generator that creates poison data similar to the normal ratings, a discriminator differentiating real and fake ratings, and an influence module appraising the effectiveness of the poison data (for the attacker's objective).

6) Evaluation Metrics: We inspect the following metrics.

- **Average predicted rating** ($\bar{r}$): We average the final ratings of the attacker's target item predicted by the poisoned Het-RecSys for each target user $\mathcal{U}_{TA}$.
- **HitRate at** 3 **(HR@**3**)**: The ratio of target users $\mathcal{U}_{TA}$ for whom the attacker's target item is ranked by the poisoned Het-RecSys in the top 3 positions among the 50 competing items $\mathcal{I}_{compete}$ [54].

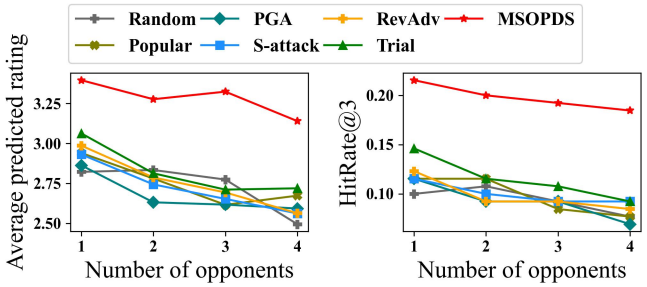
7) Hyperparameter setting: Corresponding to Algorithm 1, we set $\eta^p = 0.005$, $\eta^q = 0.05$, inner RecSys training loop number $L = 5$, and outer MSO loop number $K = 20$.

B. Result against a single opponent

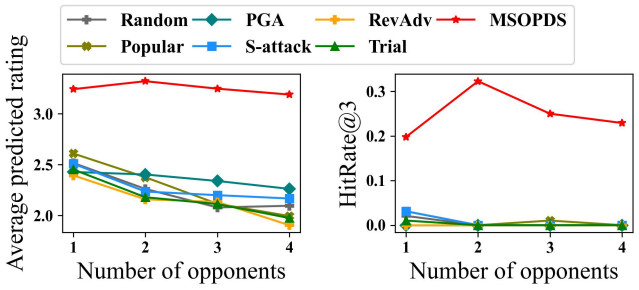
First, we evaluate MSOPDS with one attacker (leader) facing one opponent (follower), where the attacker first poisons the Het-RecSys given the original real ratings and ancillary data. Then, the opponent determines his poisoning actions (by planning a Comprehensive Attack) given both the real data and the attacker's actions (i.e., poisoning modifications on the graphs). The final prediction result is determined by the Het-

[7]We further evaluate the effect to opponent capability b_{op} in Section VI-D.

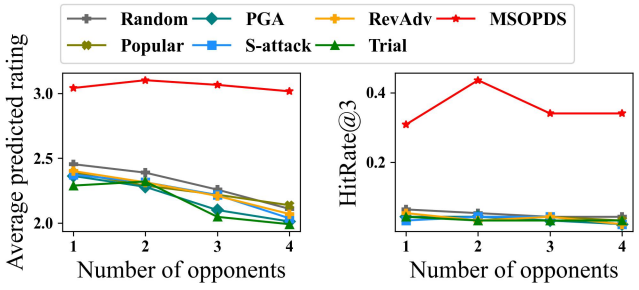
[8]For methods designed for implicit ratings, we transform rating above three stars as positive [83]. If rating for filler items are not specified, we follow [49] to produce ratings from normal distribution fitted to the true ratings.



(a) Results on the Ciao dataset.

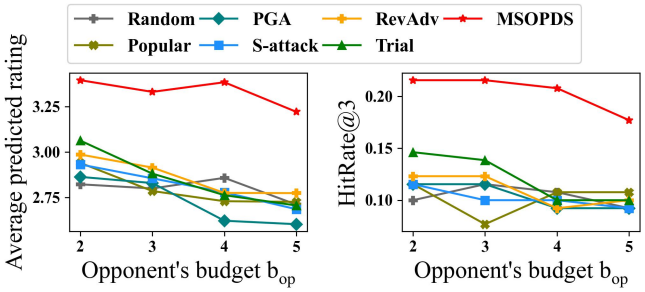


(b) Results on the Epinions dataset.

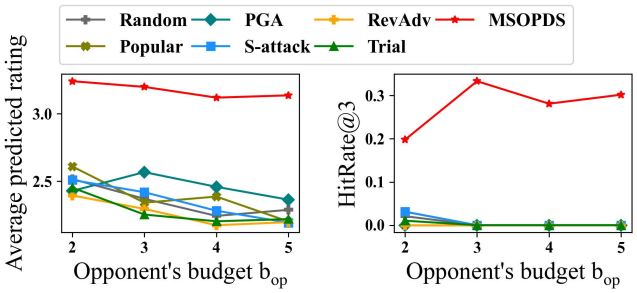


(c) Results on the LibraryThing dataset.

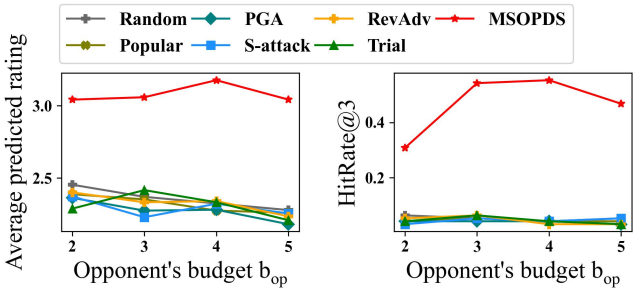
Fig. 6: Impact of the number of opponents.



(a) Results on the Ciao dataset.



(b) Results on the Epinions dataset.



(c) Results on the LibraryThing dataset.

Fig. 7: Impact of the opponent's capacity.

RecSys trained with the original data and poison data of the attacker and opponent.

Table III summarizes the results of different poisoning methods against ConsisRec [12] under different budget constraints $b \in [2, 5]$. As shown in Table III, the proposed MSOPDS under MCA yields the best performance in all budget settings and all three datasets, outperforming baseline methods by a significant margin. In particular, MSOPDS surpasses the second-best method by up to 10.6% in terms of the average predicted rating. It also consistently outperforms the second-best method in terms of the hit rate by up to 11.4%. We observe that Epinions and LibraryThing present larger differences between MSOPDS and the baselines than Ciao since Ciao has a sparser social network for the poisoning effect to propagate.

Apart from MSOPDS, all compared methods have weaker performance since they do not anticipate the poisoning attacks by opponents. Among the baselines, *PGA, S-attack, Revisit Attack*, and *Trial Attack* all plan the data poisoning attack through a bi-level optimization. However, since the above baseline and ablation methods are not designed to face the negative impact of the poisoning attacks by the opponent who acts after the attacker, these approaches suffer from bad performances and may even fall behind heuristic methods (Random Attack and Popular Attack). In contrast, the proposed MSOPDS integrates a Multilevel Stackelberg Optimization framework with Progressive Differentiable Surrogate to anticipate and evaluate the potential effects of hostile actions by the opponent. The potential opponent's actions evaluated within the Stackelberg framework play an essential role in the effective enhancement of the robustness of MSOPDS.

C. Impact of the number of opponents.

Fig. 6 compares the performance of different methods under various numbers of opponents. The opponents are assumed to sequentially conduct their respective data poisoning attacks using Comprehensive Attack, where each opponent selects real users from their customer base by BOPDS to give a 1-star rating to the target items. While all baselines do not change their attacks as the number of opponents increases (identical to the single-opponent setting), MSOPDS carefully plans the optimal poisoning attack by anticipating the opponents' actions via Multilevel Stackelberg Optimization. As a result, MSOPDS

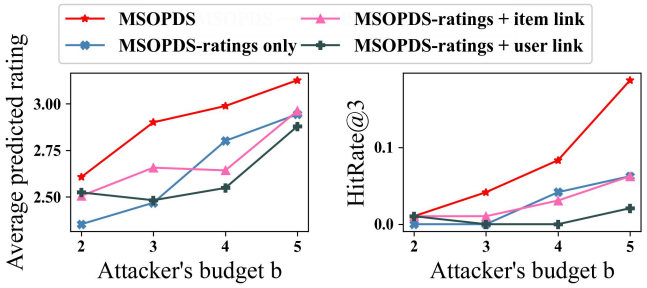


Fig. 8: Effect of different categories of poisoning actions in the Epinions dataset.

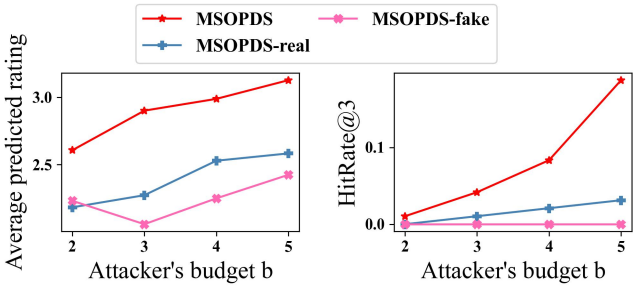


Fig. 9: Effect of creating ratings with real users and injecting ratings with fake users in the Epinions dataset.

consistently outperforms all compared baseline methods. Although the performance of baseline methods decreases drastically as the number of opponents increases, MSOPDS achieves smaller reductions in both evaluation metrics. In the Epinions dataset, MSOPDS maintains positive HitRate@3 scores, while other methods all drop to 0, manifesting the importance of modeling a multiplayer game for data poisoning attacks against RecSys.

D. Impact of the opponent's capacity

Fig. 7 compares the performance of an attacker facing a single opponent with varied budgets. While increasing the opponent's budget *reduces* the attacker's performance, MSOPDS under MCA outperforms all baselines. MSOPDS suffers from a smaller degradation than baselines in performance, since it anticipates the opponent's poisoning action plans and adjusts the attacker's poisoning action plan accordingly. Besides, the performances of the attacker's poisoning attacks in Epinions and LibraryThing are also more sensitive to the change of the opponent capacity than Ciao since these two datasets have a sparser rating record than Ciao.

E. Effect of different categories of poisoning actions

Fig. 8 compares the effectiveness of different types of poisoning actions on the Epinions dataset with **1) MSOPDS-ratings only**, where the attack does not poison the heterogeneous graph, **2) MSOPDS-ratings+item link**, which poisons the rating records and item graph, and **3) MSOPDS-ratings+user link**, which poisons the rating records and social network. As shown in Fig. 8, MSOPDS obtains the best results by carefully examining all poisoning actions. In contrast, the methods **ratings+item** and **ratings+social** both suffer a performance degradation. This is because Het-RecSys utilizes the dot product between the final user and item embeddings to determine the rating predictions [12], and poisoning only the user or only the item can only partially influence the result. Poisoning the social and item graph jointly has a compound effect because the final user and item embeddings involve a graph convolution process along the social network and the item graph. However, the convolution process dilutes the poisoning effects of modifying edges in the corresponding graph because it normalizes the neighborhood embeddings by the node degree in (15). Attacking the item graph is more effective than attacking the social network because item graph poisoning directly influences the target item's final embedding, while poisoning the social network indirectly affects the target users through creating social links between the users from the customer base $\mathcal{U}_{base}$ and the injected fake accounts.

F. Effect of poisoning by real users and fake accounts

Fig. 9 compares the effectiveness of data poisoning by real users and fake accounts with two ablation methods, including **1) MSOPDS-real**, where the attacker only hires real users, and **2) MSOPDS-fake**, where the attacker only injects the fake accounts. For a fair comparison, all methods exclude the poisoning actions against the item graph. MSOPDS obtains the best results since it carefully examines all types of poisoning actions. Besides, MSOPDS-real performs *better* than MSOPDS-fake since real users have more substantial connections in the social network. In contrast, fake users are only connected to selected real users from the customer base and thus have limited social influence. As website moderators usually detect and remove fake user accounts [86], conducting poisoning actions via real users may work better.

VII. Conclusion

In this paper, we investigate the problem of planning a data poisoning attack against RecSys formulated as a multiplayer Stackelberg game, where an attacker anticipates opponents' action plans to secure a good data poisoning results. Particularly, we target the advanced Heterogeneous RecSys, which utilizes rating records, social networks, and item graphs. We present the MSOPDS, consisting of a Multilevel Stackelberg Optimization framework and a Progressive Differentiable Surrogate model, and provide theoretical analyses. Extensive experiments demonstrate the effectiveness of MSOPDS in planning a Multiplayer Comprehensive Attack under both single-opponent and multiple-opponent settings.

Acknowledgment

This work is supported in part by NSF under grants III-1763325, III-1909323, III-2106758, and SaTC-1930941.

REFERENCES

[1] A. Jameson, M. C. Willemsen, A. Felfernig, M. d. Gemmis, P. Lops, G. Semeraro, and L. Chen, "Human decision making and recommender systems," in *Recommender systems handbook*. Springer, 2015, pp. 611–648.

[2] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems." in *IJCAI*, vol. 17. Melbourne, Australia, 2017, pp. 3203–3209.

[3] J. Tang, H. Wen, and K. Wang, "Revisiting adversarially learned injection attacks against recommender systems," in *Fourteenth ACM Conference on Recommender Systems*, 2020, pp. 318–327.

[4] L. Chen, Y. Xu, F. Xie, M. Huang, and Z. Zheng, "Data poisoning attacks on neighborhood-based recommender systems," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 6, p. e3872, 2021.

[5] Y. Deldjoo, T. D. Noia, and F. A. Merra, "A survey on adversarial recommender systems: from attack/defense strategies to generative adversarial networks," *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–38, 2021.

[6] S. W. Schmitz and M. Latzer, "Competition in b2c e-commerce: analytical issues and empirical evidence," *Electronic Markets*, vol. 12, no. 3, pp. 163–174, 2002.

[7] N. Lee. (2016, oct) Facebook's friend-based recommendations take on yelp. [Online]. Available: https://www.engadget.com/2016-10-19-facebook-recommendations.html

[8] I. Paul. (2016, oct) Facebook wants to replace yelp with your friends' recommendations. [Online]. Available: https://www.pcworld.com/article/410737/facebook-wants-to-replace-yelp-with-your-friends-recommendations.html

[9] G. Zaratti. (2021, apr) How pinterest uses machine learning to provide tailored recommendations to million of users worldwide. [Online]. Available: https://d3.harvard.edu/platform-digit/submission/how-pinterest-uses-machine-learning-to-provide-tailored-recommendations-to-million-of-users-worldwide/

[10] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, "Image-based recommendations on styles and substitutes," in *ACM SIGIR*, 2015, pp. 43–52.

[11] N. H. Center. (2023, jan) How netflix's recommendations system works. [Online]. Available: https://help.netflix.com/en/node/100639

[12] L. Yang, Z. Liu, Y. Dou, J. Ma, and P. S. Yu, "Consisrec: Enhancing gnn for social recommendation via consistent neighbor aggregation," in *ACM SIGIR*, 2021, pp. 2141–2145.

[13] B. Li, Y. Wang, A. Singh, and Y. Vorobeychik, "Data poisoning attacks on factorization-based collaborative filtering," *Advances in neural information processing systems*, vol. 29, pp. 1885–1893, 2016.

[14] H. Zhang, C. Tian, Y. Li, L. Su, N. Yang, W. X. Zhao, and J. Gao, "Data poisoning attack against recommender system using incomplete and perturbed data," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 2154–2164.

[15] K. McCormick. (2022, apr) 22 ways to ask for reviews (with copy/paste templates). [Online]. Available: https://www.wordstream.com/blog/ws/2020/07/16/how-to-ask-for-reviews

[16] S. Bernazzani. (2022, jan) How to ask & get good customer reviews [+examples]. [Online]. Available: https://blog.hubspot.com/service/get-customer-reviews

[17] A. Parsons, "Using social media to reach consumers: A content analysis of official facebook pages," *Academy of marketing studies Journal*, vol. 17, no. 2, p. 27, 2013.

[18] J. Wellemeyer. (2019, sep) This 23-year-old has made $120,000 buying and selling instagram accounts. [Online]. Available: https://www.marketwatch.com/story/this-23-year-old-has-made-120000-buying-and-selling-instagram-accounts-2019-08-12

[19] H. Britzky. (2018, jan) Nearly 50 million fake twitter accounts are being sold to real users. [Online]. Available: https://www.axios.com/2018/01/27/fake-twitter-accounts-sold-for-more-followers

[20] V. Leasure. (2022, dec) Hacker charged for attempting to make a purchase on amazon account. [Online]. Available: https://www.wxii12.com/article/alamance-hack-amazon-account-fraud-felony/42297990

[21] D. Ahmed. (2022, nov) 34 russian hacking groups stole 50 million user passwords. [Online]. Available: https://www.hackread.com/russian-hacking-groups-user-passwords/

[22] M. A. Cusumano, A. Gawer, and D. B. Yoffie, *The business of platforms: Strategy in the age of digital competition, innovation, and power*. Harper Business New York, 2019.

[23] O. C. Ferrell, M. Hartline, and B. W. Hochstein, *Marketing strategy*. Cengage Learning, 2021.

[24] A. Sinha, F. Fang, B. An, C. Kiekintveld, and M. Tambe, "Stackelberg security games: Looking beyond a decade of success." IJCAI, 2018.

[25] Z. Xiong, S. Feng, D. Niyato, P. Wang, Y. Zhang, and B. Lin, "A stackelberg game approach for sponsored content management in mobile data market with network effects," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5184–5201, 2020.

[26] J. Jin, C. Song, H. Li, K. Gai, J. Wang, and W. Zhang, "Real-time bidding with multi-agent reinforcement learning in display advertising," in *Proceedings of the 27th ACM international conference on information and knowledge management*, 2018, pp. 2193–2201.

[27] Y. Zhang, F. Rong, and Z. Wang, "Research on cold chain logistic service pricing—based on tripartite stackelberg game," *Neural Computing and Applications*, vol. 32, no. 1, pp. 213–222, 2020.

[28] L. Zhu and J. Lin, "A pricing strategy of e-commerce advertising cooperation in the stackelberg game model with different market power structure," *Algorithms*, vol. 12, no. 1, p. 24, 2019.

[29] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2847–2856.

[30] S. Hollister. (2021, sep) Amazon says it's permanently banned 600 chinese brands for review fraud. [Online]. Available: https://www.theverge.com/2021/9/17/22680269/amazon-ban-chinese-brands-review-abuse-fraud-policy

[31] D. Deahl. (2018, feb) Huawei got people to write fake reviews for an unreleased phone. [Online]. Available: https://www.theverge.com/2018/2/12/17005798/huawei-fake-reviews-best-buy-mate-10-pro-phone

[32] B. News. (2022, jun) Sony admits using fake reviewer. [Online]. Available: http://news.bbc.co.uk/2/hi/entertainment/1368666.stm

[33] J. Peters. (2021, may) Amazon-first gadget brands aukey and mpow are suddenly, suspiciously disappearing. [Online]. Available: https://www.theverge.com/2021/5/10/22428858/amazon-aukey-mpow-listings-disappearing

[34] S. Hollister. (2021, jul) Another amazon-first gadget brand has suspiciously vanished: Choetech. [Online]. Available: https://www.theverge.com/2021/7/7/22567530/amazon-choetech-removed-pulled-aukey-mpow

[35] A. Kieler. (2015, apr) Amazon files first lawsuit to block companies from selling fraudulent positive reviews. [Online]. Available: https://consumerist.com/2015/04/09/amazon-files-first-lawsuit-to-block-companies-from-selling-fraudulent-positive-reviews/

[36] L. Plaugic. (2015, feb) Yelp is suing a company for allegedly selling fake positive reviews to restaurants. [Online]. Available: https://www.theverge.com/2015/2/20/8078147/yelp-suing-company-for-fake-reviews

[37] S. Gibbons. (2022, dec) 3 e-commerce trends to watch in 2023. [Online]. Available: https://www.forbes.com/sites/serenitygibbons/2022/12/22/3-e-commerce-trends-to-watch-in-2023/?sh=4e8be85f5575

[38] Z. Nan. (2022, dec) Social commerce raises revenues. [Online]. Available: https://www.chinadaily.com.cn/a/202212/19/WS639fc1c3a31057c47eba5092.html

[39] C. Becdach. (2022, oct) Social commerce: The future of how consumers interact with brands. [Online]. Available: https://www.mckinsey.com/capabilities/growth-marketing-and-sales/our-insights/social-commerce-the-future-of-how-consumers-interact-with-brands

[40] J. Song, Z. Li, Z. Hu, Y. Wu, Z. Li, J. Li, and J. Gao, "Poisonrec: an adaptive data poisoning framework for attacking black-box recommender systems," in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 2020, pp. 157–168.

[41] S. E. Whang and J.-G. Lee, "Data collection and quality challenges for deep learning," *Proceedings of the VLDB Endowment*, vol. 13, no. 12, pp. 3429–3432, 2020.

[42] A. Balayn, C. Lofi, and G.-J. Houben, "Managing bias and unfairness in data for decision support: a survey of machine learning and data engineering approaches to identify and mitigate bias and unfairness within data management and analytics systems," *The VLDB Journal*, vol. 30, no. 5, pp. 739–768, 2021.

[43] E. Pitoura, K. Stefanidis, and G. Koutrika, "Fairness in rankings and recommenders: Models, methods and research directions," in *2021 IEEE*

37th International Conference on Data Engineering (ICDE). IEEE, 2021, pp. 2358–2361.

[44] H. Ehsan, M. A. Sharaf, and P. K. Chrysanthis, "Muve: Efficient multi-objective view recommendation for visual data exploration," in *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 2016, pp. 731–742.

[45] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE transactions on knowledge and data engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[46] Z. Tian, L. Cui, J. Liang, and S. Yu, "A comprehensive survey on poisoning attacks and countermeasures in machine learning," *ACM Computing Surveys (CSUR)*, 2022.

[47] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," *Pattern Recognition*, vol. 84, pp. 317–331, 2018.

[48] S. K. Lam and J. Riedl, "Shilling recommender systems for fun and profit," in *Proceedings of the 13th international conference on World Wide Web*, 2004, pp. 393–402.

[49] M. Fang, G. Yang, N. Z. Gong, and J. Liu, "Poisoning attacks to graph-based recommender systems," in *Proceedings of the 34th Annual Computer Security Applications Conference*, 2018, pp. 381–392.

[50] K. Christakopoulou and A. Banerjee, "Adversarial attacks on an oblivious recommender," in *Proceedings of the 13th ACM Conference on Recommender Systems*, 2019, pp. 322–330.

[51] H. Zhang, Y. Li, B. Ding, and J. Gao, "Practical data poisoning attack against next-item recommendation," in *Proceedings of The Web Conference 2020*, 2020, pp. 2458–2464.

[52] M. Fang, N. Z. Gong, and J. Liu, "Influence function based data poisoning attacks to top-n recommender systems," in *Proceedings of The Web Conference 2020*, 2020, pp. 3019–3025.

[53] H. Huang, J. Mu, N. Z. Gong, Q. Li, B. Liu, and M. Xu, "Data poisoning attacks to deep learning based recommender systems," *arXiv preprint arXiv:2101.02644*, 2021.

[54] C. Wu, D. Lian, Y. Ge, Z. Zhu, and E. Chen, "Triple adversarial learning for influence based poisoning attack in recommender systems," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 1830–1840.

[55] S. Wang, L. Hu, Y. Wang, X. He, Q. Z. Sheng, M. Orgun, L. Cao, N. Wang, F. Ricci, and P. S. Yu, "Graph learning approaches to recommender systems: A review," *arXiv preprint arXiv:2004.11718*, 2020.

[56] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 4–24, 2020.

[57] H. Dai, H. Li, T. Tian, X. Huang, L. Wang, J. Zhu, and L. Song, "Adversarial attack on graph structured data," in *ICML*. PMLR, 2018, pp. 1115–1124.

[58] J. Li, H. Zhang, Z. Han, Y. Rong, H. Cheng, and J. Huang, "Adversarial attack on community detection by hiding individuals," in *Proceedings of The Web Conference 2020*, 2020, pp. 917–927.

[59] H. Li, S. Di, Z. Li, L. Chen, and J. Cao, "Black-box adversarial attack and defense on graph neural networks," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2022, pp. 1017–1030.

[60] A. Bojchevski and S. Günnemann, "Adversarial attacks on node embeddings via graph poisoning," in *ICML*. PMLR, 2019, pp. 695–704.

[61] X. Liu, S. Si, J. Zhu, Y. Li, and C.-J. Hsieh, "A unified framework for data poisoning attack to graph-based semi-supervised learning," in *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019.

[62] P. Banerjee, L. Chu, Y. Zhang, L. V. Lakshmanan, and L. Wang, "Stealthy targeted data poisoning attack on knowledge graphs," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2021, pp. 2069–2074.

[63] Y. Zhu, Y. Lai, K. Zhao, X. Luo, M. Yuan, J. Ren, and K. Zhou, "Binarizedattack: Structural poisoning attacks to graph-based anomaly detection," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2022, pp. 14–26.

[64] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," *arXiv preprint arXiv:1205.2618*, 2012.

[65] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *The world wide web conference*, 2019, pp. 417–426.

[66] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," *arXiv preprint arXiv:1706.02263*, 2017.

[67] P. J. Bickel and K. A. Doksum, *Mathematical statistics: basic ideas and selected topics, volumes I-II package*. Chapman and Hall/CRC, 2015.

[68] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," in *ACM SIGIR*, 2020, pp. 639–648.

[69] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," *Advances in neural information processing systems*, vol. 30, 2017.

[70] R. Abraham, J. E. Marsden, and T. Ratiu, *Manifolds, tensor analysis, and applications*. Springer Science & Business Media, 2012, vol. 75.

[71] T. Fiez, B. Chasnov, and L. Ratliff, "Implicit learning dynamics in stackelberg games: Equilibria characterization, convergence analysis, and empirical study," in *ICML*. PMLR, 2020, pp. 3133–3144.

[72] R. Sato, M. Tanaka, and A. Takeda, "A gradient method for multilevel optimization," *arXiv preprint arXiv:2105.13954*, 2021.

[73] J. R. Shewchuk *et al.*, "An introduction to the conjugate gradient method without the agonizing pain," 1994.

[74] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, "Automatic differentiation in machine learning: a survey," *Journal of Marchine Learning Research*, vol. 18, pp. 1–43, 2018.

[75] H. Von Stackelberg, *Market structure and equilibrium*. Springer Science & Business Media, 2010.

[76] P. Awasthi, A. Das, and S. Gollapudi, "A convergence analysis of gradient descent on graph neural networks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 20 385–20 397, 2021.

[77] B. T. Polyak, "Gradient methods for solving equations and inequalities," *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 6, pp. 17–32, 1964.

[78] B. Bošanskỳ, S. Brânzei, K. A. Hansen, T. B. Lund, and P. B. Miltersen, "Computation of stackelberg equilibria of finite sequential games," *ACM Transactions on Economics and Computation (TEAC)*, vol. 5, no. 4, pp. 1–24, 2017.

[79] J. Tang, H. Gao, and H. Liu, "mtrust: Discerning multi-faceted trust in a connected world," in *Proceedings of the fifth ACM international conference on Web search and data mining*, 2012, pp. 93–102.

[80] T. Zhao, J. McAuley, and I. King, "Leveraging social connections to improve personalized ranking for collaborative filtering," in *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, 2014, pp. 261–270.

[81] ——, "Improving latent factor models via personalized feature projection for one class recommendation," in *Proceedings of the fifth ACM international on conference on information and knowledge management*, 2015, pp. 821–830.

[82] C. Lin, S. Chen, H. Li, Y. Xiao, L. Li, and Q. Yang, "Attacking recommender systems with augmented user profiles," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 855–864.

[83] W. X. Zhao, S. Mu, Y. Hou, Z. Lin, Y. Chen, X. Pan, K. Li, Y. Lu, H. Wang, C. Tian *et al.*, "Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 4653–4664.

[84] B. Mobasher, R. Burke, R. Bhaumik, and C. Williams, "Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness," *ACM Transactions on Internet Technology (TOIT)*, vol. 7, no. 4, pp. 23–es, 2007.

[85] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[86] M. Balaanand, N. Karthikeyan, S. Karthik, R. Varatharajan, G. Manogaran, and C. Sivaparthipan, "An enhanced graph-based semi-supervised learning algorithm to detect fake users on twitter," *The Journal of Supercomputing*, vol. 75, no. 9, pp. 6085–6105, 2019.