Manipulating Recommender Systems: A Survey of Poisoning Attacks and Countermeasures

THANH TOAN NGUYEN, Faculty of Information Technology, HUTECH University, Ho Chi Minh City, Vietnam
NGUYEN QUOC VIET HUNG, Griffith University - Gold Coast Campus, Southport, Australia
THANH TAM NGUYEN, Griffith University - Gold Coast Campus, Southport, Australia
THANH TRUNG HUYNH, Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland
THANH THI NGUYEN, School of Information Technology, Deakin University Faculty of Science Engineering and Built Environment, Waurn Ponds, Australia
MATTHIAS WEIDLICH, Humboldt-Universitat zu Berlin, Berlin, Germany
HONGZHI YIN, The University of Queensland, Saint Lucia, Australia

Recommender systems have become an integral part of online services due to their ability to help users locate specific information in a sea of data. However, existing studies show that some recommender systems are vulnerable to poisoning attacks, particularly those that involve learning schemes. A poisoning attack is where an adversary injects carefully crafted data into the process of training a model with the goal of manipulating the system's final recommendations. Based on recent advancements in artificial intelligence (AI), such attacks have gained importance recently. At present, we do not have a full and clear picture of why adversaries mount such attacks, nor do we have comprehensive knowledge of the full capacity to which such attacks can undermine a model or the impacts that might have. While numerous countermeasures to poisoning attacks have been developed, they have not yet been systematically linked to the properties of the attacks. Consequently, assessing the respective risks and potential success of mitigation strategies is difficult, if not impossible. This survey aims to fill this gap by primarily focusing on poisoning attacks and their countermeasures. This is in contrast to prior surveys that mainly focus on attacks and their detection methods. Through an exhaustive literature review, we provide a novel taxonomy for poisoning attacks, formalise its dimensions, and accordingly organise 31 attacks described in the literature. Further, we review 43 countermeasures to detect and/or prevent poisoning attacks, evaluating their effectiveness against specific types of attacks. This comprehensive survey should serve as a point of reference for protecting recommender systems against poisoning attacks. The article concludes with a discussion on open issues in the field and impactful directions for future research. A rich repository of resources associated with poisoning attacks is available at https://github.com/tamlhp/awesome-recsys-poisoning.

Authors' Contact Information: Thanh Toan Nguyen, Faculty of Information Technology, HUTECH University, Ho Chi Minh City, Vietnam; e-mail: nt.toan@hutech.edu.vn; Nguyen Quoc Viet Hung, Griffith University - Gold Coast Campus, Southport, Queensland, Australia; e-mail: quocviethung1@gmail.com; Thanh Tam Nguyen (Corresponding author), Griffith University - Gold Coast Campus, Southport, Australia; e-mail: t.nguyen19@griffith.edu.au; Thanh Trung Huynh, Ecole Polytechnique Federale de Lausanne, Lausanne, VD, Switzerland; e-mail: thanhtrunghuynh93@gmail.com; Thanh Thi Nguyen, School of Information Technology, Deakin University Faculty of Science Engineering and Built Environment, Waurn Ponds, Victoria, Australia; e-mail: thanh.nguyen@deakin.edu.au; Matthias Weidlich, Humboldt-Universitat zu Berlin, Berlin, Berlin, Germany; e-mail: matthias.weidlich@hu-berlin.de; Hongzhi Yin, The University of Queensland, Saint Lucia, Queensland, Australia; e-mail: h.yin1@uq.edu.au.

ACM 0360-0300/2024/10-ART3
https://doi.org/10.1145/3677328

CCS Concepts: • **Information systems** → **Recommender systems**; • **Security and privacy** → *Intrusion/anomaly detection and malware mitigation*; • **Computing methodologies** → *Machine learning;*

Additional Key Words and Phrases: Trustworthy recommender systems, trustworthy AI, poisoning attacks, poison attacks, model corruption

ACM Reference Format:
Thanh Toan Nguyen, Nguyen Quoc Viet Hung, Thanh Tam Nguyen, Thanh Trung Huynh, Thanh Thi Nguyen, Matthias Weidlich, and Hongzhi Yin. 2024. Manipulating Recommender Systems: A Survey of Poisoning Attacks and Countermeasures. *ACM Comput. Surv.* 57, 1, Article 3 (October 2024), 39 pages.
https://doi.org/10.1145/3677328

1 Introduction

In the era of data deluge, identifying relevant information to support decision-making has become a challenging task for the users of online services. By helping users to find useful, personalised information, a recommender system cannot only increase a service's usability, it can also contribute directly to the platform provider's ultimate success. For this reason, companies such as YouTube, Amazon, and eBay are deploying recommender systems as one of the primary means by which their customers locate items of interest such as videos, individual products, or categories of products [8]. Recommender systems have actually been around for decades; however, it has only been in the past decade that they have seen great commercial success, as evidenced by their enormous growth in recent years. In fact, back in 2018, Industry Arc forecast the recommender system market to increase from US $1.14 billion to US $12.03 billion by 2025 [1]—a prediction that looks set to prove true. Statistics like this are clear evidence that recommender systems have become an integral part of helping internet users navigate the sea of choices they face every time they go online.

In many practical scenarios, recommender systems operate in a relatively open environment. That is, these systems rely on data that is generated by other users, whether actively through comments, posts, and ratings, or passively through clicks, views, or purchases. While this openness is key to the success of many systems, given that the quality of a recommendations usually depends on drawing from a large pool of candidate data, it also renders recommender systems prone to manipulation [69, 94, 118, 154]. More specifically, recommender systems are known to be vulnerable to poisoning attacks [99], in which an adversary injects carefully crafted data into the model's training data to change the model's behaviour. Clearly, such vulnerabilities are unavoidable to some extent, even though the operators of recommender systems are generally acutely aware that such attacks can occur. Yet, poisoning attacks remain a very powerful means of manipulating users. As such, they can seriously undermine the commercial success of any company falling victim to such an attack. A prime example of a poisoning attack is to crowdsource users who will post fraudulent ratings of a company's products for a small fee per rating. Fraudulent positive ratings will promote a company's products, while fraudulent negative ones will demote those of the company's competitors. Marketplaces such as Amazon and eBay suffer from such attacks daily [70, 75, 107]. The use of fake reviews to increase the number of recommendations of particular movies is another well-documented example of a poisoning attack motivated by economic considerations [2]. Hence, a comprehensive understanding of the different types of poisoning attacks that can be launched against recommender systems—along with their goals, capacities, and impacts—is an important prerequisite for designing robust models.

1.1 Prior Classifications and Surveys of Recommender Systems

Recommender systems typically fall into one of three categories: (i) *content-based filtering*; (ii) *collaborative filtering*; or (iii) *hybrid systems*. Of these, **collaborative filtering (CF)** methods

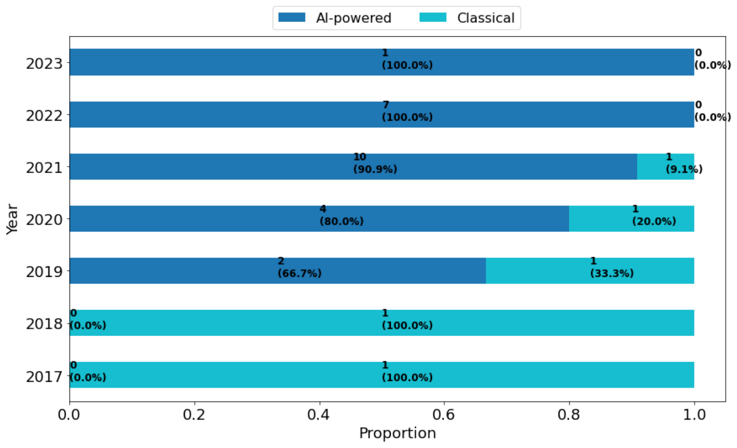

Fig. 1. The evolution of poisoning attacks over the past seven years w.r.t. the ratio of **AI-based** attacks and **classic** attacks. The two paradigms are distinguished based on the methodology followed to train the attack. The number of attacks has increased over time, with a clear shift from classic attacks to AI-based attacks.

[105] have been the mainstream of recommender system research due to the high quality of the resulting recommendations. Therefore, in this survey, we have focused on CF-based recommender systems, breaking them down into the learning scheme used [6], as follows:

— *Memory-based CF recommender systems* rely on nearest neighbour search, which is applied directly to a user's interaction history. As such, these approaches do not work with an explicit model. For example, a recommendation algorithm finds the closest users to a given target user, identifies their common preferences, and derives a recommendation for the target user based on those shared preferences.

— *Model-based CF recommender systems* presume an underlying "generative" model that explains the user-item interactions. These systems attempt to estimate a model that generates the recommendations for a given target user.

With the models adopted by CF recommender systems becoming more and more advanced over the past two decades, the attacks on them have changed as well. These attacks can be divided into two broad paradigms: AI attacks and classic heuristic attacks.

(1) *Classic heuristic attacks:* The strategy these attacks follow comprises two steps: (1) detecting malicious users, which is formulated as an optimisation problem; and (2) solving the problem, which is done through a heuristic technique. Attacks that fall into this category date back 20 years, but they are still employed these days. While many of the specific methods used by these attacks seem relatively simple, they have proven to be effective.

(2) *AI-based attacks:* Over the past seven years, we have seen the advent of various attacks that train an end-to-end framework to forge user profiles and imitate authentic user behaviours. For instance, several schemes based on generative adversarial networks (GANs) have recently been proposed that automatically mimic the behaviour of genuine users to influence the targeted system [7, 73, 115]. Another technique is to exploit the reward signal in a reinforcement learning scheme as a back-door into the recommender model [109].

As Figure 1 shows, the past seven years have seen an increasing number of attacks on recommender systems being proposed. At the same time, there has been a clear shift from classic attacks to AI-based attacks. This illustration lists the number of published attacks per year and category, as extracted using bibliometrics analysis from major computer science repositories such as Google

Table 1. The Comparison between Our Work and Existing Surveys

Category	Sub-category	[46]	[160]	[122]	[121]	[106]	[113]	[96]	Ours
Attacks	Classic heuristic attacks	✓	✓	✓	✓	✓	✓	✓	✓
	AI-based attacks	✓	✓						✓
	Unified taxonomy								✓
	Formalisation of attack dimensions								✓
	Recommender system domain of applications								✓
Countermeasures	Detection of classic heuristic attacks	✓	✓	✓	✓	✓	✓	✓	✓
	Detection of AI-based attacks	✓	✓						✓
	Formalising vulnerabilities in recommender systems								✓
	Prevention against classic heuristic attacks					✓	✓		✓
	Prevention against AI-based attacks								✓
	Countermeasures effective against attacks								✓
	Countermeasures weak against attacks								✓
Discussion	Research gaps & future research directions	✓	✓			✓			✓
	Materials collection								✓

Scholar and Scopus [42]. Details of the survey methodology are provided in Section 1.3. This trend has also been confirmed by security experts: The number of attacks that exploit AI methods can be expected to grow substantially over the coming years [3].

While AI-based poisoning attacks pose severe threats to recommender systems, existing surveys of attacks on these systems primarily focus on classic heuristic attacks and their respective countermeasures [96, 106, 113, 121, 122]. The more recent studies cover a combination of heuristic and AI-based attacks. Additionally, some surveys cover poison attacks in general [46], while others investigate these attacks in specific domains. So far, however, no one has undertaken a survey focused on recommender systems [160]. In this survey, we address this gap by providing a comprehensive overview of the state-of-the-art attacks on recommender systems and the countermeasures one can take to prevent them. As summarised in Table 1, we also go beyond existing surveys by: providing a generic taxonomy for poisoning attacks; formalising the dimensions of this taxonomy; and linking the attacks to countermeasures. This last exercise not only highlights which measures effectively detect and prevent certain attacks, it also sheds light on the ability of attacks to resist certain countermeasures. Finally, our survey is accompanied by a collection of materials that will enable scientists to kick-start their own research work in the field.

Several existing surveys consider adversarial attacks on recommender systems [43, 120]. While there are some conceptual similarities between poisoning attacks and adversarial attacks, they do have fundamental differences (as illustrated in Figure 2). The aim of an adversarial attack is to find some adversarial samples that corrupt the outcome of a recommender system at the time of inference without altering the underlying model [43, 120]. These attacks manipulate the input data to temporarily deceive the system into producing inaccurate predictions or recommendations during the attack. Poisoning attacks, however, are mounted during the model's training phase [18]. Here, the attacker injects carefully crafted data into the model's training data with the goal of fundamentally converting the *true model* into an *attack model* that yields outcomes beneficial to the attacker's goal. By injecting malicious data during training, the attacker aims to bias the system's learning process and manipulate its recommendations in a persistent manner.

To provide a more concrete example, consider an online marketplace that recommends products to users based on their browsing history. In an adversarial attack, an attacker might manipulate the content of a particular product listing or modify the user's browsing history temporarily to promote a specific item. This would influence the recommendations shown to that user during their current session. Conversely, in a poisoning attack, the attacker would inject malicious data into the system's training data, such as fraudulent reviews or manipulated purchase records. In this way, the attacker would bias the model to favour certain products or manipulate the recommendation

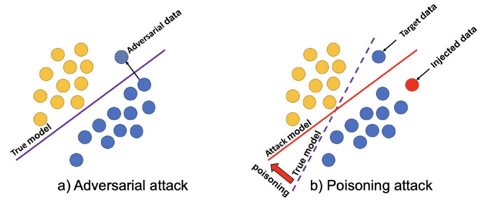

Fig. 2. Adversarial attacks vs. poisoning attacks.

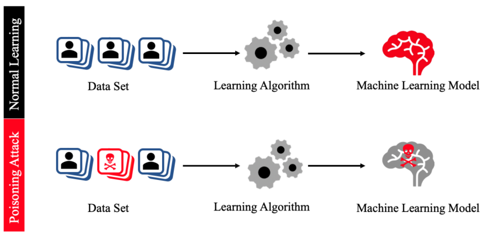

Fig. 3. The process of a typical poisoning attack.

rankings, thereby impacting the recommendations not just for a single user session but for a broader set of users over an extended period of time. Consequently, these two different types of attacks have distinct goals and implications. Although adversarial attacks are related to poisoning attacks, they are largely orthogonal to the poisoning attacks covered in this survey.

1.2 Main Contributions

This survey is intended as a point of reference for protecting recommender systems against poisoning attacks. To this end, we have structured the field of poisoning attacks and countermeasures based on a comprehensive literature review. More specifically, this survey contributes the following:

- We discuss the unique challenges that an attacker faces in the domain of recommender systems, while separating attacks on recommender systems from related approaches in fields such as machine learning and computer vision. For example, recommender systems generally exploit the correlations between user and item data to generate recommendations. These correlations can create a certain robustness in the underlying model that renders attacks difficult.
- We present the first comprehensive review of poisoning attacks on recommender systems, covering both classic heuristic modes of attack as well as AI-based attacks. To structure and present these 29 attacks, we devised a taxonomy of five dimensions that provides a holistic view of the full field of this research.
- We present an extensive review of 41 countermeasures for the identified poisoning attacks. Moreover, we link the attacks with the countermeasures, considering both measures that are effective against certain attacks as well as measures that can be expected to fail against certain attacks.
- Moving beyond our main objective, we also describe open issues in the field and conclude the survey with directions of promising future research.
- To enable researchers to kickstart their work in the field, we have assembled a public repository [4] that includes all reviewed papers as well as the program codes and the datasets released in the context of these studies. As such, this repository provides researchers entering the field with a comprehensive collection of material to support their work in securing recommender systems against potential threats.

1.3 Survey Methodology

We implemented several procedures to ensure a high-quality survey. To identify pertinent articles that establish the state-of-the-art in poisoning attacks, we first extracted papers indexed by the major computer science repositories, including Scopus and DBLP. Realising that some important work may not be indexed in these databases, we also screened articles using Google Scholar. With

DBLP, we queried key terms in the articles' titles using the repository's search interface. With Google Scholar and Scopus, we searched the articles' content. Having collected all relevant articles, we then processed and filtered them. Any paper deemed irrelevant was removed. For example, a paper was removed if it focused on poisoning attacks in a field other than recommender systems, such as machine learning or cybersecurity. We also concentrated on publications in top-tier venues (including RecSys, SIGIR, IJCAI, WWW, and KDD) and adopted a lightweight screening procedure for workshop proceedings and publications in entry-level venues.

1.4 Structure of the Survey

The rest of this article is organised as follows: Section 2 provides an overview of recommender systems and the security landscape in this field. We also highlight the unique challenges in securing recommender systems from poisoning attacks. Section 3 introduces a novel taxonomy for poisoning attacks and formally defines the dimensions of this taxonomy. In Section 4, we review existing work on poisoning attacks according to this taxonomy, covering both model-agnostic and model-intrinsic techniques. Section 5 describes countermeasures to detect and prevent poisoning attacks. Finally, we highlight research gaps and future research directions in Section 6 before concluding the article in Section 7.

2 Background

2.1 Overview of Recommender Systems

Recommender systems based on CF [105] are ubiquitous. For example, the recommender systems seen on YouTube, Netflix, Amazon, and Google Play are all CF-based systems. These models analyse the past interactions between users and items to derive a recommendation for a specific target user. Hence, CF-based recommenders can be divided into the following categories based on the underlying strategy(s) used to capture those historical interactions:

2.1.1 Matrix-factorisation-based Recommender Systems. Matrix-factorisation-based recommender systems [10, 68] assume that a small number of latent factors is sufficient to represent the users' past behaviour. Based on that assumption, a low-rank matrix is used to estimate the full user-item rating matrix. More specifically, this low-rank matrix serves as the basis for inferring missing values in the full user-item rating matrix. A recommendation for a target user is then derived as a list of the items with the highest prediction scores, even if the user has never interacted with these items before. Some methods go a step further to improve the prediction results by assigning different weights based on the activeness of the users and the popularity of the items [133].

2.1.2 Graph-based Recommender System. Graph-based recommender systems model the historic interactions between users and items as a weighted bipartite graph, called the user preference graph [57, 65]. A random walk is then performed over the user preference graph to generate the recommendations. The walk starts at a user and returns to that user with a predefined probability. The stationary probability of the consequent random walk is then used to generate the list of recommendations.

2.1.3 Association-rule-based Recommender System. The idea behind recommender systems based on association rules is to find the co-occurrence patterns in items based on the ratings issued by users [12, 17]. For instance, consider an example in which many users have assigned high ratings to two items, item X and item Y. As many users appreciate both items, the system assumes that there must be a hidden relation between the two. Hence, when a user assigns a high rating to item X, item Y will be recommended.

2.1.4 *Neighbourhood-based Recommender Systems.*

Neighbourhood-based recommender systems [92, 100] make recommendations based on the similarities between users and/or items. With *user-based* similarities, the system finds the nearest users and aggregates their ratings to recommend items. The same process applies to *item-based* similarities.

2.1.5 *Deep-learning-based Recommender Systems.*

Deep-learning-based recommender systems involve a variety of deep learning models to model the interactions between users and items [11, 86, 154]—from multilayer perceptrons [95] to autoencoders [148]; from deep reinforcement learning [90, 91, 142] to adversarial networks [31]. What all these systems have in common is that they leverage contemporary algorithms in their training schemes, which generally significantly improves the quality of recommendations compared to the other categories of recommender systems.

2.2 Poisoning Attacks

In a poisoning attack, an adversary tampers with the training data of a machine learning model to corrupt its integrity. Figure 3 illustrates the typical process of a poisoning attack compared to a normal learning process. In the latter case, a machine learning model is trained on some data that is subsequently used to derive a recommendation [78]. As such, the quality of the machine learning model depends on the quality of the data used for training. In a poisoning attack, data is injected into the training set, and hence the model, to produce unintended or harmful conclusions [93]. In this way, adversaries can launch an attack against even the most advanced training algorithms and the most complex models.

In a poisoning attack on a recommender system, the data injected into the training set will typically relate to fake users and their fake ratings as an attempt to modify the resulting recommendations. Here, the usual goal is to either promote an item (if bolstering one's own reputation) or demote one (if attacking one's competitors) [155]. The general course of action for an adversary is to infiltrate the recommender system by registering a number of fake users. These users will then rate a subset of items to coerce the desired result. Independent of the attack strategy, and no matter whether it is a classic heuristic or an AI-based attack, the fake data, which can be crafted either manually or automatically, will influence any model that learns from the data. As a result, the recommendations derived from the model will be manipulated towards the adversary's goal.

One way to categorise poisoning attacks on recommender systems is to divide them by the type of recommender system they are designed to target. Are they *model-agnostic*? Or *model-intrinsic*? Model-agnostic attacks ignore the underlying model and any algorithms used to build it. For this reason, the effectiveness of these types of attacks is typically limited. Model-intrinsic attacks, however, are optimised for a specific type of training process. As such, these attacks can cause substantial damage to the underlying model.

Some scholars have likened poisoning attacks to *profile pollution* attacks, e.g., Reference [84]. However, there are some notable differences between the two:

— *Profile pollution attacks* are designed to disrupt the rating behaviour of regular users with the intention of compromising the system. These attacks can not only be targeted at recommendation systems but also at other personalised online services, such as information retrieval or web search systems. For instance, attackers can manipulate a user's browsing history to distort their personalised recommendations. However, executing such attacks usually requires that the adversary exploit vulnerabilities in the **cross-site request forgery (CSRF)** [20, 77], which limits their scalability.
— *Poisoning attacks*, as detailed above, inject fake users together with fake ratings into a system so the model learns biased behaviour [33]. Not only can these attacks be performed at large scales, adversaries can also incorporate multiple different goals into their attacks.

In thinking about these two types of attacks, we note that poisoning attacks pose a more severe threat to economies and society. From an economic perspective, the motivations to promote/demote products and services on a large scale are very strong. Such attacks can make or break a company. However, they can cause irreversible harm to the fairness and trustworthiness of the targeted recommender system and bring the platform owner into disrepute. From a social perspective, poisoning attacks can manipulate popular belief. For instance, an adversary that compromises a recommender system delivering online news can directly manipulate a community's opinions about anything, including an election or public policy.

2.3 Characteristics of Poisoning Attacks on Recommender Systems

It is important to note that poisoning attacks are not simply malicious assaults on an online system. In fact, poisoning attacks are of great significance to the sustainability of machine learning models in that they are the *de facto* standard procedure for evaluating a model's robustness against noise or polluted data. Considering the vital role poisoning attacks play in the field of machine learning, they have been studied for a wide range of machine learning models, including support vector machines [116], decision trees [50], regression models [23], and neural networks [55]. However, generic poisoning attacks on machine learning models have only limited application to recommender systems for several reasons.

— *Data correlation.* While machine learning models learn hidden knowledge from one source of training data, recommender systems learn user preferences from the interactions of two data sources—the users and the items. Hence, a recommender system's robustness stems from the correlations between these data [97]. This makes poisoning attacks on recommender systems different from attacks of traditional machine learning models, most particularly, in that they typically require more effort to execute. For instance, in a poisoning attack on a computer vision system, it may be enough to change a single pixel [111], whereas a successful poisoning attack on a recommender system would require the adversary to inject many, many user-item correlations into the model's dataset.

— *The lack of prior knowledge.* Another popular approach to poisoning attacks in the field of machine learning is to leverage gradient descent to discover dedicated perturbations, which are subsequently combined with the regular samples. As these combinations are undetectable, they can significantly affect the quality of the learned models. By contrast, recommender systems are typically black-box systems that do not provide access to the underlying model. This means the attack must usually be based only on the training data (i.e., the rating matrix). Additionally, users of recommender systems often have privacy concerns and, hence, are hesitant to publish their preferences. This means that an attack will commonly be based on only a small subset of the data on which the recommender model was trained.

— *Multiple attack goals.* When attacking a recommender system, an adversary typically has several attack goals in mind. For example, one aim might be to promote a set of items, while another goal is to tarnish the reputation of their competitors' items. However, fusing multiple goals generally involves a tradeoff of some sort, since certain actions that are beneficial to one goal may undermine the success of the other goal. Additionally, attacks striving to achieve multiple goals are generally easier to characterise and, hence, to detect.

2.4 Challenges when Securing Recommender Systems against Poisoning Attacks

The general problem of securing machine learning models against poisoning attacks has been studied extensively [53, 99, 99, 159]. While the literature includes elaborate countermeasures against poisoning attacks [14], challenges remain, especially when trying to secure a recommender system against such an attacks.

— *Openness.* Recommender systems are typically public, i.e., they are accessible to large numbers of users, making them very vulnerable to poisoning attacks. The data used to influence the recommendations is also open, in the sense that it cannot be characterised *a priori* in terms of volume or distribution, which offers many degrees of freedom for data manipulation.
— *Concept drift.* To secure recommender systems, many existing methods apply anomaly detection techniques to identify and filter out fake users. However, recommender systems commonly suffer from a phenomenon called concept drift, meaning that user behaviour constantly evolves due to seasonal or trending preferences. Consequently, real users can easily be misclassified as fake users [52].
— *Imbalanced data.* Any classification of regular and fake users is also hampered by the imbalance of the respective classes. Attacks typically have a certain "budget," meaning there is a limit on how many fake users/ratings an adversary can inject into the system and, moreover, this number of fake users is often only a tiny portion of the overall user base.

2.5 Application Perspectives of Poison Attacks in Recommender Systems

In this survey, we examine the key domains where recommender systems are widely used: e-commerce [87], social media [158], and news recommendations [140]. In e-commerce [87], poison attacks can cause biased recommendations, erode user trust, and result in financial losses. Social media platforms [158] face the risks of misinformation, polarisation, and the promotion of harmful content. News recommendations [140] can be manipulated to shape opinions and compromise democratic values. Understanding these vulnerabilities helps to develop targeted defences to protect users, maintain trust, and ensure the responsible use of recommender systems across domains.

3 Threat Model and Taxonomy

Tabassi et al. [114] proposed a general taxonomy for organising poisoning attacks on machine learning models. However, in the light of the above attack characteristics and challenges faced in the context of recommender systems, we present a novel taxonomy that is specifically geared towards poisoning attacks on recommender systems. As illustrated in Figure 4, our taxonomy comprises five distinct dimensions. These five dimensions were included for the following reasons:

(1) The dimensions are specific to the domain of recommender systems. For example, attacks in the general field of machine learning may seek to violate the integrity or confidentiality of the learned models. Yet, in the context of recommender systems, the goals specifically relate to promoting or demoting items.
(2) The dimensions are relevant to the general threat model of data poisoning attacks. These dimensions are important for highlighting the current gaps in research and for outlining directions of future research into poisoning attacks on recommender systems. See Section 6 for a more detailed discussion on future directions of research.

Notably, this taxonomy applies to both classic heuristic attacks and AI-based attacks. The remaining subsections provide a formal and comprehensive view of the dimensions of the taxonomy and of the threat model associated with poisoning attacks on recommender systems (Section 3.2–Section 3.6). The section begins with some general notions and notations related to CF-based recommender systems as a primer for those not familiar with this material (Section 3.1).

3.1 Collaborative Filtering: A Primer

Consider a basic CF scenario, where $\mathcal{U}$ is the number of users and $\mathcal{I}$ is the number of items. The tuple $(\langle u, i\rangle, r_{ui})$ indicates that user u has rated the item i with a rating score of r_{ui}. The rating matrix $\mathcal{R} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$ captures all these interactions between users and items. Note that

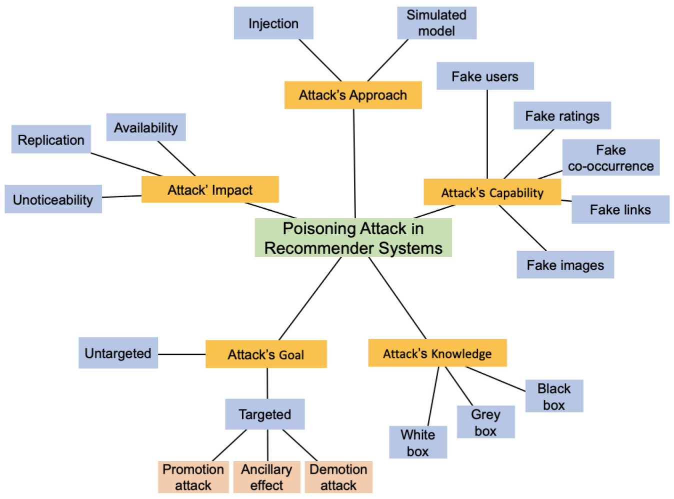

Fig. 4. Taxonomy of poisoning attacks on recommender systems.

not all entries of the rating matrix $\mathcal{R}$ are actually filled with rating scores. Indeed, the goal of the recommender system is to estimate the missing entries. This estimation task can be formulated as predicting the complete user-item matrix $\tilde{\mathcal{R}}$ based on prior knowledge of $\mathcal{R}$, where each $\tilde{r}_{ui} \in \tilde{\mathcal{R}}$ represents a prediction of the rating score r_{ui}. The prediction problem is then formulated as a predictive model to approximate the function: $f : \mathcal{U} \times \mathcal{I} \rightarrow \mathbb{R}$ with $f : \langle u, i \rangle \mapsto \tilde{r}_{ui}$, where θ is the set of model parameters. Before revealing the optimisation problem for this recommendation task, we need to introduce the standard loss functions commonly used in recommender systems.

Common Loss Functions in Recommender Systems. In general, loss functions in recommender systems are a means to measuring how close the predicted rating is $\tilde{r}_{ui}$ to the actual rating r_{ui}. The two most common loss functions are:

(1) *Least squares loss:* which is the maximum likelihood estimation under a Gaussian distribution [101]. More specifically, the loss function casts the recommendation problem as a regression problem [62], formally defined as

$$\ell(r_{ui}, \tilde{r}_{ui}) = \sum_{r_{ui} \in \mathcal{R}, \tilde{r}_{ui} \in \tilde{\mathcal{R}}} w_{ui}(r_{ui} - \tilde{r}_{ui})^2, \tag{1}$$

where w_{ui} is the contributed weight of the user u-item i pair into the overall loss.

(2) *Binary cross-entropy loss:* which is the maximum likelihood estimation under a Bernoulli distribution [40]. More precisely, the loss function casts the recommendation problem as a binary logistic regression problem [83], formally defined as

$$\ell(r_{ui}, \tilde{r}_{ui}) = \sum_{r_{ui} \in \mathcal{R}, \tilde{r}_{ui} \in \tilde{\mathcal{R}}} r_{ui} \log P_{ui} + (1 - r_{ui}) \log(1 - Pui), \tag{2}$$

where P_{ui} is the non-linear activation over the predicted rating $\tilde{r}_{ui}$. This is then calculated by $P_{ui} = \sigma(\tilde{r}_{ui}) = \frac{1}{1+e^{-\tilde{r}_{ui}}}$.

The Optimisation Problem. Having chosen a loss function (ℓ), the prediction task is then formulated as an optimisation problem:

$$\min_{\theta} \sum_{r_{ui} \in \mathcal{R}} \ell(f(\langle u, i \rangle; \theta), r_{ui}). \tag{3}$$

The predicted matrix $\tilde{\mathcal{R}}$ is then used to recommend a list of items to a user that the user has not encountered yet. For instance, if the goal is to suggest K items to user u, then the model will extract the top-K items that: (i) the user has not yet rated; and (ii) have the highest predicted score in the row vector $(\tilde{r}_{u1}, \tilde{r}_{u2}, \ldots, \tilde{r}_{uN})$ of $\tilde{\mathcal{R}}$.

3.2 The Adversary's Goal

Generally, attacks on recommender systems aim to either promote an item, demote an item, or both. However, an attack can either be *non-targeted*, where the goal is a general degrading of performance; or *targeted*, where a specific item or items is the quarry. These attack goals are formally characterised next.

3.2.1 Untargeted Poisoning Attacks. The goal in an untargeted attack is to maximise the error of the recommender system, eventually rendering it useless. Suppose $\mathcal{U}'$ is the set of controlled users injected by an adversary with $\mathcal{R}' \in \mathbb{R}^{|\mathcal{U}'|\times|\mathcal{I}|}$ being their rating scores. An untargeted poisoning attack would be formulated as the following optimisation problem:

$$\min_{\mathcal{R}'} ||\mathcal{R}'|| \quad s.t : f(\langle u, i\rangle; \theta) \neq r_{ui}, \forall\, r_{ui} \in \mathcal{R} \cup \mathcal{R}'. \tag{4}$$

Alternatively, relying on a maximisation problem instead of the more common CF minimisation (Equation (3)), the adversary will seek to maximise the loss between the predicted and actual rating scores. This leads to the following problem formulation:

$$\min_{\mathcal{R}'} ||\mathcal{R}'|| \max_{r_{ui} \in \mathcal{R}\cup\mathcal{R}'} \ell(f(\langle u, i\rangle; \theta), r_{ui}). \tag{5}$$

3.2.2 Targeted Poisoning Attacks. The adversary's goal in a targeted attack is to increase or decrease the popularity of a targeted item. These attacks can be referred to as *promotion* and *demotion* attacks, respectively. Since both attacks are similar and exchangeable, we have only described a promotion attack to keep the discussion concise. For a demotion attack, simply change instances of "maximise" to "minimise" and vice versa.

Let t be the target item, and let $\mathcal{R}'$ be the rating scores of the controlled users injected by adversary. A promotion attack would then be formulated as follows:

$$\min_{\mathcal{R}'} ||\mathcal{R}'|| \quad s.t : f(\langle u, i\rangle; \theta) = r_{ut}, \forall\, r_{ui} \in \mathcal{R} \cup \mathcal{R}'. \tag{6}$$

Here, the attack mechanism tries to boost the visibility of the target t and, thus, minimise the loss between the predicted and the expected rating score r_{ut} of all users who rated item t. This problem is similar to the primary setting of CF (Equation (3)), except that only the target item is involved.

$$\min_{\mathcal{R}'} ||\mathcal{R}'|| \min_{r_{ut} \in \mathcal{R}\cup\mathcal{R}'} \ell(f(\langle u, t\rangle; \theta), r_{ut}) \tag{7}$$

While the aim of the promotion attack in Equation (7) is to boost the popularity of the target item, this type of attack is oftentimes referred to as a *non-resilience* targeted attack. The reason is that this kind of attack is easily detected due to the level of skew in the rating distribution, as the model will excessively favour the target item to the exclusion of all else. Conversely, a *resilience* targeted attack strives to promote the target less obviously and without hampering the recommendations for any other items in the system. This attack is formulated as an unconstrained optimisation problem, as follows:

$$\min_{\mathcal{R}'} ||\mathcal{R}'|| \min_{r \in \mathcal{R}\cup\mathcal{R}'} \ell(f(\langle u, i\rangle; \theta), r_{ui}) + \ell(f(\langle u, t\rangle; \theta), r_{ut}). \tag{8}$$

Beyond promotion and demotion attacks, a couple of studies have recently shed light on a novel attack objective called the *ancillary effect* [96]. These approaches are hybrid attacks that aim to manipulate a group of users or items.

3.3 The Adversary's Knowledge

The characteristics of the attacker's knowledge play an essential role in the threat model. This is because the impact of an attack will differ significantly, depending on the extent of knowledge an attacker has about the system they are trying to undermine. Consider the following types of attacks based on the background knowledge an adversary might have:

— *Black-box attack.* In a black-box attack, the adversary does not know the details of the target system. Specifically, they will not be aware of the architecture of the system, the function f used for prediction or its parameters θ, or the users' past behaviours $\mathcal{R}$.

— *Grey-box attack.* In a grey-box attack, the adversary has limited knowledge. As such, she can merely modify the user-item interaction matrix $\mathcal{R}$ by injecting a limited number of controlled users along with their ratings to create a poisoned matrix $\mathcal{R}'$.

— *White-box attack.* In this type of attack, the adversary has thorough knowledge of the system, including the prediction function f, its parameter set θ, and the entire history of interactions between users and items $\mathcal{R}$.

In machine learning, both white-box and black-box attacks are widespread, and both have been shown to be efficient attacks during the training phase. In the field of recommender systems, however, *grey-box attacks* are of particular importance, as will be discussed in more detail in Section 4.

3.4 The Attack Impact

Considering the different types of background knowledge possessed by an adversary, there are three general, long-term impacts of a poisoning attack on a recommender system, as listed below.

— *Availability.* Similar to other learning-based algorithms, recommender systems make decisions based on the data they have amassed in the past. White-box and grey-box attacks disturb the input data to these algorithms [64]. At first, an adversary will inject fake users to manipulate the recommender system and weaken the accuracy of the underlying model. However, this weakened model will ultimately serve as a backdoor, helping the adversary to obtain complete control over the system, thereby compromising its availability.

— *Replication.* While black-box attacks are less efficient at harming the system's availability [15], they may instead replicate the system. A common tactic in a black-box attack is to reverse-engineer the underlying model as a simulation. Once known, the model can then be replicated and exploited.

— *Unnoticeability.* Some poisoning attacks ensure that their approach goes unnoticed by preserving crucial data characteristics when injecting fake data. Conventional techniques to guard the recommender system generally fail to address this particular vulnerability. Even with protection techniques in place, such undetected attacks can often be effective and may cause significant damage.

3.5 The Adversary's Capabilities

When attacking a recommender system in the real world, an adversary has many tools at her disposal. Commonly, the attacker will possess at least one of the following capabilities:

— *Fake users.* Any recommender system can be manipulated if the attacker injects enough fake users. However, in practice, there is a tradeoff between the ability of an attacker to execute an attack and a system's robustness. In particular, injecting a large number of fake users is difficult from an operational point of view. Many injections tend to inevitably yield markers that separate controlled users from ordinary ones, so the injected users can be detected easily. The maximum number of fake users that can be injected as part of a specific attack is called *attack size*, denoted as $\mathcal{U}'$. This number is typically much lower than the overall number of users in the system (introduced above as $\mathcal{U}$).

— *Fake ratings.* For each fake user, a number of non-zero fake ratings is injected into the target model. However, there is typically a bound k on the number of ratings that can be injected, called the *profile size.* This is because, in real-life, users generally only interact with a few items in a the system [88]. Formally, the fake ratings are denoted as $\mathcal{R}'$. Moreover, the domain for these ratings is given by $\mathcal{B}(\mathcal{R}, k)$, which denotes the space around the actual rating matrix $\mathcal{R}$ of radius k, i.e., the maximum number of amended ratings.

— *Fake co-occurrence.* Instead of inserting fake ratings for fake users directly, an attacker may also manipulate the recommender model by introducing fake co-occurrences between items. That is, given a target item, the poisoning attack is based on injecting visits to other items for users that are already linked to the target item.

— *Fake links.* Several approaches for recommender systems exploit knowledge graphs as an auxiliary source of information to improve recommendation accuracy [129]. As these knowledge graphs frequently depend on third-party data, they represent a vulnerability. That is, an adversary may add fake links to rewrite the knowledge graph's structure, thereby influencing the recommender system.

— *Fake images.* Many recommender systems use images of items to address the cold-start problem. Again, such external data sources provide an angle through which to attack the recommender system [76]. An attacker may create dedicated images of items that, once injected into the recommender system, promote particular items and increase their ranking in the given recommendations.

3.6 The Attack Approach

To achieve their goal, the approach taken by adversaries often depends on their knowledge and capabilities. In general, the field makes a distinction between the following two types of approaches:

— *Injection.* The most common scenario for a poisoning attack is that the adversary has limited knowledge about the targeted system. As such, the adversary will tend to inject a limited number of well-designed users. In addition, these fake users will also assign ratings to several other items to disguise the attack and their real target.

— *Simulation.* In a black-box setting, where the adversary does not have enough knowledge to perform an injection, the attack will usually be based on simulating the targeted recommender system. More precisely, a surrogate model is trained using data collected from the recommender system to reproduce the targeted system's behaviour.

4 Poisoning Attacks

Poisoning attacks in recommender systems can be divided into two groups: (1) *model-agnostic attacks*, which can be executed to evaluate the trustworthiness of any recommender system; and (2) *model-intrinsic attacks,* which target a specific type of recommender system. Table 2 provides a summary of the surveyed attacks.

4.1 Model-agnostic Poisoning Attacks

4.1.1 Attack Formulation. A model-agnostic poisoning attack can be executed against any recommender system, regardless of its underlying algorithm. This attack strategy involves injecting manipulated data into the training set, which causes the framework to learn a biased model. The objective here is to manipulate the recommender system into promoting/demoting specific items. Mathematically, a model-agnostic poisoning attack can be formulated as follows:

$$\min ||\mathcal{R}' - \mathcal{R}||_2^2 \quad s.t. : r'_{ui} \in \mathcal{R}', \forall \text{ fake user } u \text{ and items } i. \tag{9}$$

This formulation considers the original rating matrix, denoted as $\mathcal{R}$, as well as the perturbed rating matrix, denoted as $\mathcal{R}'$. The rating r'_{ui} is the score for an item i that the attacker plans to inject

Table 2. Summary of Poisoning Attacks

Name	Authors	Type	Attack Goal				Knowledge & Capability								Approach & Impact				
			Untar. Attack	Tar. Attack			Adversary's Knowledge			Adversary's Capability					Attack Appr.		Attack Impact		
				Promotion Attack	Demotion Attack	Ancillary Effect	White box	Grey box	Black box	Fake Users	Fake Ratings	Fake co-occurrence	Fake Links	Fake Images	Injection	Simulation	Availability	Replication	Unnoticeability
	Model-agnostic																		
MA-01	Lam et al. [70]	Classic	✓	✓	✓			✓		✓	✓				✓		✓		✓
MA-02	Song et al. [109]	AI-based		✓					✓	✓	✓				✓	✓		✓	
MA-03	Tang et al. [115]	AI-based	✓	✓	✓			✓		✓	✓				✓	✓		✓	
MA-04	Lin et al. [73]	AI-based		✓				✓		✓	✓				✓		✓		✓
MA-05	Wu et al. [124]	AI-based		✓			✓	✓		✓	✓				✓		✓		✓
MA-06	Zhang et al. [158]	AI-based	✓	✓					✓	✓	✓				✓	✓		✓	
MA-07	Fan et al. [47]	AI-based		✓					✓	✓	✓				✓	✓		✓	✓
MA-08	Barbieri et al. [19]	AI-based		✓				✓		✓	✓				✓		✓		
MA-09	Wu et al. [127]	AI-based		✓				✓				✓			✓		✓		
MA-10	Chen et al. [35]	AI-based		✓					✓	✓	✓				✓		✓		
MA-11	Zhang et al. [150]	AI-based		✓					✓	✓	✓					✓		✓	✓
MA-12	Lin et al. [74]	AI-based		✓	✓				✓	✓	✓				✓		✓		✓
	Model-intrinsic																		
MI-01	Li et al. [71]	Classic	✓	✓	✓		✓			✓	✓				✓		✓		✓
MI-02	Yang et al. [134]	Classic		✓	✓		✓	✓	✓			✓			✓		✓		
MI-03	Fang et al. [49]	Classic		✓			✓	✓	✓	✓	✓				✓		✓		✓
MI-04	Chris et al. [38]	AI-based		✓			✓			✓	✓				✓		✓		✓
MI-05	Hu et al. [61]	AI-based		✓						✓	✓				✓		✓		
MI-06	Chen et al. [34]	Classic	✓	✓			✓			✓	✓				✓		✓		
MI-07	Zhang et al. [149]	AI-based		✓		✓			✓	✓	✓				✓	✓		✓	
MI-08	Fang et al. [48]	Classic		✓			✓	✓		✓	✓				✓		✓		✓
MI-09	Chen et al. [36]	Classic		✓				✓	✓	✓	✓				✓		✓		
MI-10	Zhang et al. [151]	AI-based		✓				✓		✓	✓				✓	✓		✓	
MI-11	Yue et al. [144]	AI-based		✓					✓	✓	✓				✓	✓		✓	
MI-12	Wu et al. [129]	AI-based		✓			✓						✓		✓		✓		
MI-13	Liu et al. [76]	AI-based		✓			✓	✓	✓					✓	✓		✓		✓
MI-14	Huang et al. [63]	AI-based		✓				✓		✓	✓				✓		✓		✓
MI-15	Zhang et al. [155]	AI-based		✓			✓			✓	✓				✓		✓		✓
MI-16	Rong et al. [98]	AI-based		✓					✓	✓	✓				✓		✓		
MI-17	Wu et al. [126]	AI-based	✓						✓	✓	✓				✓		✓		✓
MI-18	Yi et al. [140]	AI-based	✓					✓		✓	✓				✓		✓		✓
MI-19	Nguyen et al. [87]	AI-based		✓				✓		✓	✓				✓	✓		✓	✓

on behalf of a fictitious user u. The notation $|| \cdot ||_2^2$ denotes the L2 norm. To ensure the injected rating r'_{ui} is a valid rating appearing in the perturbed rating matrix $\mathcal{R}'$, the constraint $r'_{ui} \in \mathcal{R}'$ is needed. This constraint reflects the attacker's objective of presenting the injected data as legitimate, thereby deceiving the recommender system into considering it as genuine information.

4.1.2 Related Work. Many model-agnostic poisoning attacks were originally designed to test the general robustness of a recommender system in terms of its trustworthiness. As such, these attacks are designed to be independent of any specific prediction model or class of such models. Some prominent examples of these model-agnostic attacks follow (Table 3).

Lam et al.'s shilling attack [70] pioneered the formal definition of a poisoning attack. This attack focuses on promoting specific items in the system without disclosing self-interest (**MA-01** in Table 2). This classic approach assumes the adversary has knowledge of the rating distributions, and so fake users with manipulated ratings are injected into the training set. Although they are hard to detect, shilling attacks can ultimately corrupt the recommendation model. By contrast, Song et al.'s PoisonRec [109] (**MA-02**) is an adaptive poisoning attack with reinforcement learning as the training scheme. By actively injecting interactions by fake users into the system, PoisonRec dynamically improves its own attack strategies through a reward signal. Additionally, a hierarchical tree structure guides the search space sampling process, which optimises the attack's efficiency.

Table 3. Domain and Evaluation Comparison

Name	Authors	Year	Type		Evaluation Metrics															Domains	Datasets
			explicit	implicit	Pre@k	RMSE	MAE	AR	NI	UI	HR@k	ASR	Recnum	DR	NDCG@k	Agr@k	ER@k	AUC	MRR		
	Agnostic																				
MA-01	Lam et al. [70]	2004	✓				✓													movie	ML
MA-02	Song et al. [109]	2020		✓									✓							game, movie, phone, clothing	St, AMV, ML
MA-03	Tang et al. [115]	2020	✓								✓									POI, location-based	GOW
MA-04	Lin et al. [73]	2020	✓								✓									movie, automotive	ML, FT, AAT
MA-05	Wu et al. [124]	2021	✓								✓				✓					movie	ML, FT
MA-06	Zhang et al. [158]	2021		✓	✓						✓									movie, book, digital music, social network, citation network	ML, AMB, ADM, NF, TW, G+, CIT
MA-07	Fan et al. [47]	2021		✓							✓				✓					movie	ML, NF
MA-08	Barbieri et al. [19]	2021	✓								✓									movie	ML
MA-09	Wu et al. [127]	2021	✓		✓						✓									movie, product	DB, CI
MA-10	Chen et al. [35]	2022	✓								✓				✓					movie, book, music	ML, BC, LA
MA-11	Zhang et al. [150]	2022	✓								✓									product, game, POI, location-based	ABT, St, GOW
MA-12	Lin et al. [74]	2022	✓								✓									movie, POI, location-based, automotive, tool, grocery, app	ML, FT, YE, AAT, THI, GGF, AA
	Instrinsic																				
MI-01	Li et al. [71]	2016	✓			✓		✓												movie	ML
MI-02	Yang et al. [134]	2017		✓					✓	✓										video, product, movie, POI, location, social network	YT, eB, AMV, YE, LI
MI-03	Fang et al. [49]	2018	✓								✓									movie	ML, AMV
MI-04	Chris et al. [38]	2019	✓								✓									movie	ML
MI-05	Hu et al. [61]	2019	✓									✓								movie	FT
MI-06	Chen et al. [34]	2019	✓			✓														movie	NF,ML
MI-07	Zhang et al. [149]	2020	✓											✓						product	ABT
MI-08	Fang et al. [48]	2020	✓								✓									tourism, music	YE, ADM
MI-09	Chen et al. [36]	2021	✓								✓									movie	FT, ML, AMV
MI-10	Zhang et al. [151]	2021		✓							✓									movie	ML, AIV
MI-11	Yue et al. [144]	2021	✓								✓				✓	✓				movie, product	ML, ABT
MI-12	Wu et al. [129]	2021		✓							✓									movie, business	ML, FTr
MI-13	Liu et al. [76]	2021		✓							✓									clothing, fashion	AMM, TC
MI-14	Huang et al. [63]	2021	✓								✓									movie, music	ML, LA
MI-15	Zhang et al. [155]	2022	✓								✓						✓			movie, phone	ML, AMP
MI-16	Rong et al. [98]	2022	✓								✓						✓			movie, music	ML, ADM
MI-17	Wu et al. [126]	2022	✓								✓				✓					movie, product	ML, ABT
MI-18	Yi et al. [140]	2022		✓											✓			✓	✓	news	MIND, Feed
MI-19	Nguyen et al. [87]	2023	✓		✓						✓							✓		app, movie, phone, music	FR, ML, AMP, LA

Tang et al.'s adversarially learned an injection attack [115] (**MA-03**) as an alternative approach. It focuses on automatically learning the behaviour of fake users from a separate model. This method offers an exact solution for generating fake user behaviour and includes scalability schemes for large-scale datasets. While these attacks aim to manipulate the recommendation system, their intent is not to replicate the system entirely, but rather to influence its outputs. Lin et al.'s GAN-based approach [73] (**MA-04**) combines GANs with the **Augmented Shilling Attack framework (AUSH)** to allow tailored attacks based on specific budgets and goals. The approach, which has demonstrated some robustness against various defence strategies, has been tested on a variety of recommender systems. By contrast, Zhang et al.'s black-box attacks [47] (**MA-07**) and [158] (**MA-06**) leverage surrogate models and clone user profiles to promote items in targeted domains. Using deep reinforcement learning, the model's parameters are adjusted by reward signals.

Wu et al.'s TrialAttack [124] (**MA-05**) employs triple adversarial learning and a flexible end-to-end framework to generate malicious users. By formulating an optimisation problem, TrialAttack creates fake users with different intents, effectively imitating a natural distribution of actual user ratings. Barbieri et al.'s approach [19] (**MA-08**) uses variational autoencoders to approximate real user profiles, which tends to generate more realistic malicious profiles. Wu et al.'s GOAT [127] (**MA-09**) integrates GANs with **graph neural networks (GNNs)** to strike a balance between feasibility and effectiveness, where high-order relations are modelled between co-rated items. Chen et al.'s KGAttack [35] (**MA-10**) incorporates knowledge graphs into a hierarchical policy network to generate fake user profiles. Zhang et al.'s LOKI [150] (**MA-11**) focuses on complex black-box next-item recommender systems, using a recommender simulator and a result estimator to guide the training of attack agents . Finally, Lin et al.'s Leg-UP [74] (**MA-11**) uses GANs to discover patterns in user behaviour and generate fraudulent user profiles, addressing visibility concerns in conventional shilling attacks.

4.2 Model-intrinsic Poisoning Attacks

Model-intrinsic attacks are designed and optimised for a specific type of recommender system. Here, the different attacks are therefore grouped according to the system type.

4.2.1 Attack Formulation. In contrast to model-agnostic poisoning attacks, model-intrinsic poisoning attacks are based on knowledge about the underlying algorithms of the target recommender system. The adversary optimises an objective function to inject manipulated data while maintaining recommendation accuracy but simultaneously minimising the chances of detection. Formally, let $\mathcal{L}$ represent the original loss function used in the collaborative filtering process, which typically measures the squared difference between the predicted ratings and the actual ratings. Conversely, let $\mathcal{A}$ denote the poison attack objective function designed to maximise the impact of any injected poisoned data while minimising the effect of that data on genuine interactions. To encompass both functions, an overall objective function can be defined as a combination of $\mathcal{L}$ and $\mathcal{A}$, as follows:

$$\mathcal{E} = \alpha * \mathcal{L} + \beta * \mathcal{A}. \tag{10}$$

This formulation introduces two weighting coefficients, α and β, to control the balance between accuracy and the impact of the attack. These coefficients determine the tradeoff between these two factors. The aim of the optimisation process then becomes one of finding the optimal values for the latent factor matrices of the users and items. Additionally, the injected poison data is also optimised to minimise the objective function $\mathcal{E}$.

4.2.2 Matrix-factorisation-based Recommender Systems. Li et al. (**MI-01**) [71] pioneered poisoning attacks and demonstrated the effectiveness of generating undetected fake data. By contrast, Yang et al. [134] (**MI-02**) focused on injecting fake item co-occurrence data into the training set. So, while Li et al. seek to manipulate user behaviour, Yang et al. look to exploit patterns of item selection. The former essentially imitates normal user behaviour, while the latter leverages the co-occurrence of items. In another vein, Christakopoulou et al. [38] (**MI-04**) introduced a "learning-to-attack" model that is based on a repeated general-sum game. This strategy focusses on the dynamic aspects of the attack, whereas Hu et al.'s [61] (**MI-05**) attack was devised specifically for social recommender systems. Their approach is to manipulate users through fake social connections, which highlights the adaptability to social RSs, even without direct connections to the attacker.

Chen et al. [34]) (**MI-06**) proposed a poisoning attack for cross-domain recommendation. They addressed the challenges of cross-domain recommendations through a dual-level optimisation problem and an approximation scheme. Zhang et al. [149] (**MI-07**), however, developed LOKI, a black-box attack that relies on deep reinforcement learning. LOKI focuses on the next-top-k recommendations and overcomes access restrictions through training on a recommender simulator. Fang et al. [48] (**MI-08**) target influential users in recommender systems based on matrix factorisation to optimise ratings. Their approach is essentially to recommend a target item to regular users. Zhang et al. [151] (**MI-10**), instead, focus on handling data incompleteness in recommender systems by using a probabilistic generative model to select less perturbed users and items. Liu et al. [76] (**MI-13**) address the cold-start problem in top-k recommender systems by exploiting images through their **Adversarial Item Promotion (AIP)** attack. By contrast, Yue et al. [144] (**MI-11**) developed black-box attacks on sequential recommender systems by extracting model parameters without accessing user-item interaction data.

4.2.3 Graph-based Recommender Systems. Several poisoning attack methods have been developed for graph-based recommendation systems. For example, Fang et al.'s attack [49] (**MI-03**) injects fake users and rating scores into the recommender system to manipulate the model. The premise is to optimise the fake rating scores to maximise impact while minimising the chances of being detected. Wu et al.'s attack [129] (**MI-12**) targets recommender systems that rely on knowledge graphs, using deep reinforcement learning to choose the optimal attack combinations in a step-by-step manner. The knowledge graph is manipulated before the model is trained, which effectively alters the target item's ranking. Last, GSPAttack [87] (**MI-19**) incorporates generative surrogate-based techniques to fabricate users and user-item interactions, yet recommendation accuracy is maintained by preserving the correlations in the data. Unlike the other methods, GSPAttack emphasises the importance of maintaining high accuracy for non-target items, which is crucial for the success of the poisoning attack. These attacks demonstrate different approaches and considerations in poisoning graph-based recommender systems, showcasing optimisation-based, reinforcement learning-based, and surrogate-based techniques for manipulating recommender systems.

4.2.4 Neighbourhood-based Recommender Systems. Poisoning attacks targeting neighbourhood-based recommender systems are discussed at some length in Chen et al. [36] (**MI-09**). The presented framework, known as UNAttack, shares similarities with other model-intrinsic poisoning attacks by strategically introducing fabricated users into the target models. This ensures that the recommended items are prioritised for a large number of regular users. Additionally, this research reveals two key findings: (i) recommender systems relying on Euclidean-based similarity measures demonstrate high resilience against such attacks; and (ii) UNAttack can be successfully applied to other recommender systems, including neural CF and Bayesian personalised ranking systems.

4.2.5 Deep-learning-based Recommender Systems. Huang et al. [63] (**MI-14**) devised a poisoning attack specifically for these deep learning–based recommender systems. The attack involves injecting ratings made by fabricated users into the model. Notably, the definition of these ratings is formulated as an optimisation problem, which can be approximately solved by incorporating numerous heuristic rules. Interestingly, the attack remains effective even when the attackers only have access to a relatively small fraction of user-item interactions.

4.2.6 Federated Recommender Systems. PipAttack [155] and FedAttack [126] are two notable poisoning attacks designed for federated recommender systems. PipAttack [155] (**MI-15**) exploits the popularity bias in data-driven recommender systems, creating fake users that mimic popular items to increase the likelihood that they will be included in recommendation lists. Similarly, Rong et al. [98] (**MI-16**) put forward some poisoning attacks for the federated learning setting that do not require prior knowledge. Rather, they use random approximation and hard user mining to approximate benign user embeddings. Wu et al.'s [126] FedAttack (**MI-17**) is an untargeted attack strategy that employs hard negative sampling. These attacks degrade the performance of the victim model while managing to evade most existing defence and detection mechanisms. Additionally, Yi et al. [140] (**MI-18**) introduce UA-FedRec, the first study on untargeted attacks in a federated news recommendation system. UA-FedRec undermines the performance of the global model in federated learning for news recommendation with minimal malicious client involvement.

4.3 Comparison of Poisoning Attacks

Our goal in this section is to provide a comprehensive comparison of the characteristics of poisoning attacks as found in the literature. Our five characteristics are: the adversary's goal, the impact, the approach, the adversary's capability, and their knowledge. Additionally, we have summarised the studies in terms of the domain(s) considered, the interaction types included in the attack, the evaluation metrics used, and the related datasets.

4.3.1 The Adversary's Goal. Although the literature includes research on both targeted and untargeted poisoning attacks, targeted attacks dominate the field (see Table 2). All approaches support targeted attacks that promote or demote specific items, but only a small portion of the published strategies consider untargeted attacks, which seek to reduce the efficiency of a recommender system in general. This imbalance is not surprising, given the much clearer incentives associated with promotion/demotion attacks. As a result, adversaries have paid much more attention to targeted attacks and, accordingly, so have scholars. Similarly, only a few works discuss the ancillary aim of manipulating a group of users or items.

4.3.2 The Attack Impact. Turning to the impact of poisoning attacks, we infer from Table 2 that two-thirds of the attacks focus on the availability of the targeted system, and one-third focus on both attacking and replicating the victim model. While the number of studies related to replication attacks is much lower than that of availability attacks, the impact caused by replication attacks is much more severe. The reason being that, in addition to manipulating the target system to perform as desired by the adversary, replication attacks devise a clone of the victim model, which could be deployed in direct competition with the target recommender system [67].

4.3.3 The Attack Approach. As for the approach followed in a poisoning attack, all approaches rely, in some form, on *injecting* fake users and ratings into the training data. Yet, Table 2 highlights that several attacks additionally rely on *simulating* the targeted recommender system, mostly to derive user profiles for injection.

4.3.4 The Adversary's Knowledge. Attacks can also be categorised according to the level of knowledge the adversary has of targeted system, i.e., white-box attacks, grey-box attacks, and black-box attacks. Notably, the total rate of attacks in Table 2 does not sum to 100%, since certain attacks support multiple levels of knowledge. Nonetheless, the table does illustrate that the execution rate is evenly distributed among the three levels.

4.3.5 The Adversary's Capabilities. The adversary's capabilities refers to their ability to inject different types of data into the training set. The types include: (i) fake users; (ii) fake ratings; (iii) fake co-occurrences; (iv) fake links; and (v) fake images. Among the discussed attacks, the most prevalent ones involve injecting fake users and ratings into the training data. For the majority of studies, this tactic has been the focus. Fake co-occurrences are considered in a smaller proportion of attacks, while fake links and fake images represent specialised attack vectors, each employed exclusively by a single attack.

4.3.6 Domains. The literature on poison attacks in recommender systems covers a broad range of application domains, including movies [98, 126, 155], POI and location-based services [74, 115, 134, 150], citation networks [158], news [140], and others. These domains demonstrate the extensive impact of poison attacks, as they can manipulate user preferences, influence market trends, promote malicious applications, manipulate consumer choices in fashion, and disrupt information dissemination across various sectors. The prevalence of poison attacks across so many domains underscores the importance of developing robust countermeasures to safeguard the integrity and trustworthiness of the online data we all rely on these days.

4.3.7 Interaction Types. Another relevant dimension is the type of interaction the system uses to generate its recommendations. A distinction can be made between the attacks designed to target recommender systems that rely on *explicit* interactions between users and items and those where these interactions are *implicit*. Table 9 illustrates that both types of interactions are covered in the literature, although the main focus falls on recommender systems with explicit interactions.

4.3.8 Evaluation Metrics. Researchers have used a range of evaluation metrics to assess the efficacy of various poisoning attacks [13]. An overview of these metrics appears in Table 4, along

Table 4. Commonly Used Evaluation Metrics

Abbrv	Term
Pre@k	The fraction of the top-K recommended items falling into the recommended ones
RMSE	Root mean square error
MAE	Mean absolute error
AR	Average rating for specific items
NI	The increased ranks of the targeted item
UI	User impression/Item popularity
HR@k	The hit rate ratio
ASR	The attack success rate
Recnum	The number of page view that contains target items
DR	The average display rate
nDCG@k	Normalised discounted cumulative gain
Agr@k	Agreement at rank k
ER@k	The exposure rate at rank k
AUC	Area under the ROC Curve
MRR	Mean Reciprocal Rank

Table 5. Commonly Used Datasets

Abbrv	Name	Abbrv	Name
YT	YouTube	AMB	Amazon Book
eB	Ebay	NF	Netflix
AMV	Amazon Movie	TW	Twitter
YE	Yelp	G+	Google+
LI	Linkin	CIT	Citation Network
ML	MovieLens	FTr	Fund Transaction
FT	Film Trust	DB	Douban
St	Steam	CI	Ciao
GOW	Gowalla	AMM	Amazon Men
ABT	Amazon Beauty	TC	Tradesy
AAT	Amazon Automotive	LA	Last.fm
ADM	Amazon Digital Music	AMP	Amazon Cell-phone
AIV	Amazon Instant Video	BC	Book-Crossing
THI	Tool and Home Improvement	AA	Apps for Android
GGF	Grocery and Gourmet Food	MIND	Online News
FR	App Recommendation	Feeds	Online News

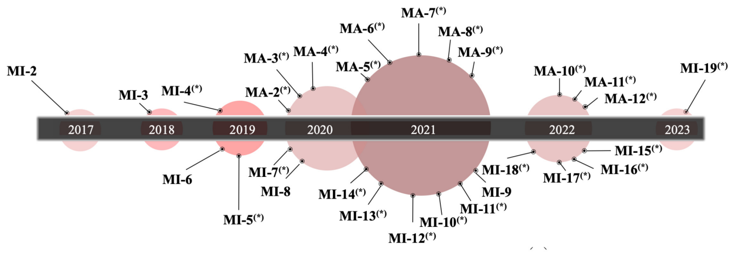

Fig. 5. The evolution of studies on poisoning attacks over time. $^{(*)}$ indicates AI-based methods.

with a link to the corresponding attack most commonly being evaluated (see Table 2). Among these metrics, *HR@k* stands out as the most predominant, being adopted by nearly two-thirds of studies. This dominance is not surprising, considering that *HR@k* is commonly used as the primary evaluation metric [104]. The second most prominent metric is *NDCG@k*, employed in a smaller portion of the studies. Other metrics are less popular and have only been used in one or two studies.

4.3.9 Datasets. Table 5 provides an overview of contemporary datasets commonly used to evaluate poisoning attacks on recommender systems. The table includes links to the respective studies, which are listed in Table 2. Among the popular datasets, the *MovieLens* dataset has been used in over two-thirds of the studies, while datasets from various product categories of Amazon have been used in half the studies. Both the MovieLens and Amazon datasets are widely recognised within this research community [41], making them valuable starting points. Three studies have used the Netflix datasets to help develop their attacks; two have used Yelp; while datasets derived from other public websites, such as Twitter, Google+, YouTube, and so on, have been used sporadically.

4.4 Summary

Finally, Figure 5 illustrates that poisoning attacks have attracted a great deal of attention recently. The timeline indicates the growth in related studies over the past seven years. Here, the early studies, i.e., those published prior to 2018, laid the foundations of research on poisoning attacks designed for recommender systems—most of which adopted classic, heuristic techniques. However, the advance of AI in recent years has led to considerable growth in the overall number of published studies, most of them indeed being AI-based (especially over the period 2020–2023).

5 Countermeasures

Countermeasures to poisoning attacks can be divided into two subgroups: (1) *detection methods*, i.e., approaches that aim to identify user profiles that have been created in poisoning attacks; and (2) *prevention methods*, i.e., approaches that aim to make a recommender system more robust to poisoning attacks, without explicitly trying to identify manipulated profiles. In this section, we review both types of methods. We then map which countermeasures are effective against which types of poisoning attacks in (Section 5.3) as well as which countermeasures cannot be expected to work well against certain attacks (Section 5.4). Finally, we summarise our insights in Section 5.5.

5.1 Detection Methods

This section begins with a discussion on the kinds of features that are commonly used to detect poisoning attacks. This is followed by a review of the traits most often employed by detection methods as signals for detecting poison attacks. The section concludes with a discussion on the actual methods that have been proposed to differentiate between genuine user profiles and those that have been used as part of an attack.

5.1.1 *Detection Features.*

Existing detection methods can rely on either *model-agnostic* features or *model-intrinsic* features.

Model-agnostic Features. Model-agnostic features are designed to capture general abnormal behaviour, independent of the type of attack model being used. These include:

(1) ***Rating deviation from mean agreement (RDMA):*** This is the average deviation in ratings given by a particular user to a number of specific items compared to other users, weighted by the inverse rating frequency of those items. Formally, let r_i^u be the rating given by a user u to the item i, and let $\overline{r_i}$ be the average of the ratings given to item i. n_{r^i} represents the number of ratings given to item i, while n_u denotes the number of items that user u has rated. Thus, the feature is defined as

$$RDMA_u = \frac{\sum_{i=0}^{n_u} \frac{|r_u^i - \overline{r^i}|}{n_{r^i}}}{n_u}. \tag{11}$$

(2) ***Weighted deviation from mean agreement (WDMA):*** This feature is the weighted variance of RDMA, which captures the cumulative differences in user ratings for sparse items. Using the same notation as above, WDMA is formulated as:

$$WDMA_u = \frac{\sum_{i=0}^{n_u} \frac{|r_u^i - \overline{r^i}|}{n_{r^i}^2}}{n_u}. \tag{12}$$

(3) ***Length variance (LengthVar):*** This is a measure of the difference in profile length, which is the number of items rated by that profile. It is calculated against the average length of all profiles. With n_u as the length of the profile u, and $\overline{n_u}$ as the average length of all profiles. LengthVar is defined as:

$$LengthVar_u = \frac{n_u - \overline{n_u}}{\sum_{u \in U} (n_u - \overline{n_u})^2}. \tag{13}$$

(4) ***Degree of similarity with top neighbours (DegSim):*** This feature reflects the average similarity of a user to its the top-k neighbours. Here, k denotes the number of neighbours, while $sim_{u,v}$ is the similarity between users u and v, computed using, say, Pearson's

correlation [103]. The feature is then formulated as

$$DegSim_u = \frac{\sum_{v=1}^{k} sum_{u,v}}{k}. \tag{14}$$

Model-intrinsic Features. Despite the many benefits of model-agnostic features, it has been proven they are not particularly good at differentiating between genuine and malicious user profiles [113], especially when a real user exhibits some unusual behaviour. To address this issue, various model-intrinsic features have been proposed. Below, we review some common, representative notions associated with such features:

(1) ***Mean variance (MeanVar)***: This is a measure that partitions malicious profiles into different constituent parts: extreme ratings (for targeted items); other ratings (items to simply fill up the profile, i.e., so-called filler items); and unrated items. This feature is estimated by computing the mean variance between “other ratings” and the average rating of all items. Let $P_{u,F}$ denote the filler items F of a user u, and let $r_{u,j}$ be the rating for item j given by user u. $\overline{r_u}$ is the mean rating given by user u to items. MeanVar is then defined as

$$MeanVar = \frac{\sum_{j \in P_{u,F}(r_{u,j} \times \overline{r_u})^2}}{|P_{u,F}|}. \tag{15}$$

(2) ***Filler mean target difference model (FMTS):*** This is a measure to assess the degree of difference between the ratings in the targeted partition and those in the filler partition. Adopting the above notation, this feature is formulated as

$$FMTD = \left| \frac{\sum_{i \in P_{u,F}} r_{u,i}}{|P_{u,T}|} - \frac{\sum_{k \in P_{u,F}} r_{u,k}}{|P_{u,F}|} \right|. \tag{16}$$

(3) ***Filler average correlation (FAC):*** This feature reflects the correlation between the ratings in a profile compared to the average rating of all items. It is defined as

$$FAC = \frac{\sum_{i \in I_u} (r_{u,i} - \overline{r_i})}{\sqrt{\sum_{i \in I_u} (r_{u,i} - \overline{r_i})^2}}. \tag{17}$$

(4) ***Filler mean difference (FMD):*** This feature measures the average value of the absolute difference of the profile’s ratings and the average rating of all items, defined as follows:

$$FMD = \frac{1}{U_u} \sum_{i=1}^{|U|} |r_{u,i} - \overline{r_i}|. \tag{18}$$

5.1.2 Detection Traits. This subsection extends the foundational framework of detection traits developed by Sundar et al. [113]. Their framework classifies attack detection traits into four groups: *(i) user profiles*, *(ii) target ratings*, *(iii) filler ratings*, and *(iv) side information*.

User profile. This first group of traits is used by detection methods to differentiate between malicious profiles and genuine profiles. We have added new characteristics for comparison:

(1) *Similarity:* The more similar a profile is to its neighbours, the higher the probability that the profile has been created as part of an attack.
(2) *Size:* The attack size (i.e., the number of injected profiles). A size of highly similar profiles much smaller than the entire set of user profiles indicates an attack.
(3) *Group behaviour:* Commonly, user behaviour has hidden characteristics. For example, a group of malicious users might have a positive correlation in rating variance. Such group behaviour reveals valuable clues to help detect malicious users.

(4) *Attributes:* The user attributes of genuine and injected profiles will tend to follow different distributions. For this reason, statistical analysis methods can often reveal abnormal profiles to help detect attacks.

Target Rating. Recall that target ratings are the ratings given to the item that attackers aim to promote or demote. Two traits related to target ratings need to be considered:

(1) *Crowdability:* The frequency of ratings of a targeted item is generally abnormally high following a poisoning attack. These peaks in rating frequency can be an indicator of attack.
(2) *Skewed Ratings:* The ultimate goal of the adversary is to manipulate user attitudes toward a targeted item. Hence, fake ratings will tend to deviate from average ratings—a finding that can be exploited to detect attacks.

Filler Rating. As mentioned above, filler ratings are ratings assigned to regular items as part of an attack rather than to the target item. Two traits need to be considered here as well.

(1) *Rating:* To disguise an attack by maximising similarity, the ratings assigned to filler items will usually be close to the current average rating. Based on this observation, a good strategy may be to first assess the ratings of filler items and then identify the adversary.
(2) *Length:* The number of items rated by a profile is known as the length of the profile. Given that an adversary will try to create a profile that is as similar to a genuine profile as possible, a malicious profile will usually be much longer than a regular profile. Profile length can therefore give away an attacker.

Side Information. More recently, poisoning attacks have begun to rely on side information to manipulate the target system [110]. This opens up another class of traits for consideration.

(1) *Co-occurrence:* Some recommender systems use co-occurrence graphs as the basis for suggesting items to users [134]. With these systems, an adversary might inject crafted co-occurrence information into the recommender system to manipulate the recommendations.
(2) *User-user graph:* The similarity between users is also a characteristic to differentiate malicious from genuine users. Using graph mining algorithms [152], the similarity of users can be explored by computing the similarity between nodes of a graph. The resulting graph of user interactions provides a valuable angle from which to detect an attack.

5.1.3 Overview of Detection Methods. Table 6 provides a summary of methods used to detect poisoning attacks. These methods are reviewed in more detail below, starting with the supervised methods, before turning to the semi-supervised and unsupervised ones.

Supervised Methods. In supervised settings, there needs to be a label to be able to distinguish a malicious profile from a genuine profile. To our best knowledge, Chirita et al. [37] (**CM-1**) were the first to formulate detecting a poisoning attack as a classification problem. Their model verifies success using the RDMA and DegSim features to detect malicious profiles. Additionally, to improve the performance of the classifier, two abstract features increase the generalisability of the mode. However, the main issue with using abstract features is that regular users with unusual behaviours might be misclassified as malicious. Therefore, several research teams, i.e., References [26] (**CM-2**) and [27] (**CM-3**), have suggested including some more specific features to improve accuracy. This improved model has subsequently been used to detect a wide range of attacks, including average, segment, random, and bandwagon attacks.

Williams et al. [85, 123] (**CM-4**) further improved the performance of classifiers by combining a reverse-engineered attack with a mechanism that detects anomalous ratings. These studies go on to confirm that using various features in an aggregated manner, such as RDMA, WDMA, LengthVar, MeanVar, DegSim, FAC, FMD, and FMTD, mean the attacks can be detected more quickly. The main weakness of this approach is that the detection model's performance strongly depends on

Table 6. Overview of Methods to Detect Poisoning Attacks on Recommender Systems

Name	Authors	Year	Detecting Features																Detecting Traits									
			Model-agnostic Features											Model-intrinsic Features					User Profiles				Filter Ratings		Target Rating		Side Information	
			RDMA	WDMA	WDA	LengthVar	DegSim	DegSim'	Hv-score	Entropy	Latent	Tracking	Trust	MeanVar	FMTD	FAC	FDM	Latent	Similarity	Size	Group behaviour	Attributes	Length	Ratings	Crowdability	Skew Ratings	Co-occurrence	User-user graph
	Agnostic																											
CM-01	Chirita et al. [37]	2005		✓	✓																			✓		✓		
CM-02	Burke et al. [26]	2006		✓	✓	✓								✓	✓									✓		✓		
CM-03	Mobasher et al. [27]	2006	✓			✓	✓							✓						✓				✓				
CM-04	Williams et al. [85, 123]	2007	✓	✓		✓	✓							✓	✓	✓	✓		✓					✓		✓		
CM-05	Zhang et al. [146]	2012	✓	✓	✓	✓	✓							✓		✓	✓		✓	✓								
CM-06	Zhang et al. [147]	2014										✓												✓		✓		
CM-07	Zhou et al. [166]	2016												✓	✓	✓	✓							✓	✓	✓		
CM-08	Yang et al. [138]	2016	✓	✓	✓	✓								✓	✓	✓							✓	✓				
CM-09	Hao et al. [60]	2019									✓													✓	✓	✓		✓
CM-10	Xu et al. [132]	2019											✓								✓			✓		✓		
CM-11	Zhou et al. [161]	2020									✓													✓		✓		
	Semi-supervised																											
CM-12	Wu et al. [128]	2012	✓	✓	✓	✓	✓			✓				✓	✓									✓		✓		
CM-13	Cap et al. [32]	2013	✓			✓	✓			✓					✓									✓		✓		
	Instrinsic																											
CM-14	Zhang et al. [153]	2006										✓												✓		✓		
CM-15	Mehta et al. [79]	2007	✓	✓															✓					✓		✓		
CM-16	Bryan et al. [25]	2008							✓															✓		✓		
CM-17	Meta et al. [82]	2009	✓		✓		✓				✓								✓					✓		✓		
CM-18	Bhaumik et al. [21]	2011	✓	✓	✓				✓										✓	✓								
CM-19	Chung et al. [39]	2013									✓													✓		✓		
CM-20	Bilge et al. [22]	2014									✓								✓									
CM-21	Zhou et al. [162]	2014	✓					✓											✓					✓	✓	✓		
CM-22	Zhang et al. [157]	2015									✓													✓		✓		
CM-23	Zhou et al. [164]	2015	✓				✓	✓																	✓	✓		
CM-24	Xia et al. [130]	2015	✓																					✓		✓		
CM-25	Yang et al. [135]	2016																	✓									✓
CM-26	Yang et al. [136]	2017	✓	✓	✓	✓	✓							✓	✓	✓	✓		✓					✓		✓		
CM-27	Zhang et al. [145]	2018									✓													✓		✓		
CM-28	Zhou et al. [165]	2018									✓	✓									✓			✓		✓		
CM-29	Cai et al. [28]	2019										✓							✓					✓		✓		
CM-30	Cai et al. [29]	2019									✓													✓		✓		
CM-31	Cai et al. [30]	2019									✓										✓							
CM-32	Aktukmark et al. [9]	2019									✓											✓		✓		✓		
CM-33	Yang et al. [137]	2020									✓													✓		✓	✓	
CM-34	Hao et al. [59]	2021									✓													✓		✓		✓
CM-35	You et al. [141]	2023																✓	✓			✓						✓

the classifier's choices. Hence, the authors recommend using a **support vector machine (SVM)** to optimise accuracy. Other approaches, such as Zhang et al. [146] (**CM-5**) and Zhang et al. [147] (**CM-6**), incorporate meta-learning over the set of features used in **CM-4**. This strategy renders the approach more effective compared to SVM classifiers that simply rely on single or majority voting.

Another problem is class imbalance. To overcome this issue, Zhou et al. [166] (**CM-7**) devised a dual-phase detection method, called SVM-TIA. The method proceeds in two steps: (1) the Borderline-SMOTE technique [108] is used to obtain initial results while alleviating any class imbalances; and (2) these results are then fine-tuned and analysed to discover malicious profiles. The approach mostly relies on model-intrinsic features, including FMTD, FMD, FAC, and MeanVar. Yang et al. [138] (**CM-8**) followed a similar direction to handle class imbalances, also using a two-step process to improve detection accuracy. In the first phase, a statistical analysis of various attack models is applied to extract features from user profiles. Then, RAdaBoost, a variant of AdaBoost, efficiently classifies the injected profiles. Hao et al. [60] (**CM-9**) developed an ensemble of detection methods that goes beyond prior approaches by considering the "novelty of items." Here, the popularity of each item is calculated based on features extracted from the ratings and a user graph, where differences in popularity provide hints as to the targeted items.

Relying on trust features and time series analysis, Xu et al. [132] (**CM-10**) developed TSA-TF, which is a detection method designed specifically for social recommender systems. Here, suspicious items are first detected by employing a single exponential smoothing technique that results in a set of suspicious profiles. Second, four features are extracted based on rating patterns and the trust relations between users. Finally, an SVM classifies these extracted features to reveal malicious profiles. To overcome the limitations of hand-crafted features, Zhou et al. [161] (**CM-11**) devised DL-DRA, an approach to detecting poison attacks based on deep learning. The authors propose an end-to-end learning process that learns directly from raw rating data. More precisely, a bicubic interpolation algorithm [44] scales down the rating matrix to overcome problems with sparsity. A **convolutional neural network (CNN)** then extracts any hidden user features based on the resized rating matrix. Finally, these hidden features are parsed through an algorithm designed specifically to detect poisoning attacks.

Semi-supervised Methods. In most recommendation systems, the number of labelled users is typically quite small. Further, annotating labels for the remaining users is generally impractical, as it costs too much and, moreover, access to the data is usually restricted. In these situations, semi-supervised methods have proven to be quite effective. For example, Wu et al. [128] (**CM-12**) proposed the first semi-supervised framework for detecting poisoning attacks called HySAD. HySAD uses MC-Relief, a wrapper that selects features by aggregating various popular attack detection metrics. The authors employ a **semi-supervised Naïve Bayes (SNB_λ)** [131] classifier to categorise both labelled and unlabelled profiles. Similarly, Cao et al. [32] (**CM-13**) introduced a semi-supervised algorithm, called Semi-SAD, that can learn from both labelled and unlabelled user profiles. This work combines the Naïve Bayes classifier with a variant of the **expectation-maximisation algorithm (EM-λ)** in a sequential manner. Specifically, Semi-SAD initially trains the Naïve Bayes classifier using a small set of labelled profiles. The results are then fine-tuned and improved by incorporating unlabelled profiles with the EM-λ algorithm.

Unsupervised Methods. In the complete absence of labels, the only alternative is an unsupervised method of detection. The first to propose such an approach was Zhang et al. [153] (**CM-14**). Their method builds on the idea that the distribution of ratings over time might reveal various kinds of hidden attacks. On this premise, the authors grouped consecutive ratings into windows of the same size k and found, through a theoretical proof, that some specific window sizes are optimal for detecting an attack, but only if the number of malicious profiles is known. For circumstances when the number is not known, they devised a heuristic algorithm to select the window size dynamically.

To improve detection accuracy, Mehta et al. [79, 82] applied **principal component analysis (PCA)** to the problem of profile detection (**CM-17**). Bryan et al.'s [25] algorithm called UnRAP was inspired by the efficacy of a technique developed in the field of gene expression analysis called Hv_score [24, 25]) (**CM-16**). The novelty of their approach is that they formulate detecting an attack as a problem of detecting anomalous structures. The strategy is so successful that the method can detect some types of attacks with higher confidence than even supervised methods.

Based on the hypothesis that malicious profiles will be similar to each other, many approaches employ clustering to separate malicious profiles from genuine profiles, i.e., References [21] (**CM-18**), [22] (**CM-20**), [136] (**CM-26**), [145] (**CM-27**). For example, Chung et al. [39] (**CM-19**) applied the Beta distribution algorithm [89] to detect poisoned profiles. Wen et al. [162] (**CM-21**) put forward De-TIA, which, rather than only considering individual attacks, detects groups of attacks. The central idea of De-TIA is to incorporate both the DegSim metric [163] and the RDMA metric [164] into a single framework to better differentiate between malicious profiles and genuine profiles. The authors also extend their work to the problem of targeted rating pattern analysis. Unlike prior work, which only tends to address a specific type of attack, Zhang et al. [157] (**CM-22**) continuously scan the recommender system to identify attacks as they emerge. Their method looks for the propagation of fraudulent actions; it also estimates the reliability of users in propagation-based systems.

Adopting a statistical approach, Zhou et al. [164] (**CM-23**) developed a method of identifying the ratings patterns associated with malicious profiles and their features. Likewise, Xia et al. [130] focused on detecting anomalous items directly to filter out items manipulated by fake profiles (**CM-24**). Separating malicious profiles from genuine profiles can also be done through user-user graphs and correlation analysis [135] (**CM-25**).

Zhou et al. [165] (**CM-28**) detect malicious behaviour by analysing ratings as a time series. In this way, they are able to identify anomalous user groups. Similar ideas are discussed in Cai et al. [28] (**CM-29**). Their approach, called BS-SC, starts with an in-depth analysis of user behaviour, after which, the hidden features are extracted to discriminate between fake users and genuine users. The second step is to build a similarity matrix behaviour for all the profiles. Cai et al. [29] (**CM-30**) presented a similar approach that involves analysing user behaviour. Unlike most unsupervised methods, which only consider rating distributions to detect malicious profiles, Aktukmak et al. [9] (**CM-32**) rely on user features as an additional reference to enhance detection performance. More specifically, these researchers employ a probabilistic factorisation model to project these two data sources into a latent space. Based on the learned latent space, the framework provides a detection mechanism for new users based on anomaly statistics. In addition to protecting recommender systems from poisoning attacks, Cai et al. [30] (**CM-31**) proposed an approach called **Value-based Neighbour Selection (VNS)** that also improves the profitability of e-retailers at the same time.

Last, Yang et al. [137] (**CM-33**) published a detection method called IMIA-HCRF that trains models based on various aspects of the users' behaviour. It incorporates the density of user ratings as well as co-visitation behaviour to identify malicious profiles. Exploiting recent advances in deep learning with graphs, Hao et al. [59] (**CM-34**) present a method that involves reconstructing user-user graphs that are subsequently used to identify malicious profiles. Similarly, You et al. [141] (**CM-35**) propose Anti-FakeU, a framework that integrates confidence scores from a malicious user detector into a neighbourhood aggregation mechanism. Fake users, generated automatically, eliminate the need for labels. The approach involves constructing a user-user graph to capture malicious behaviour patterns and a novel GNN-based detector to identify fake users.

5.2 Prevention Methods

5.2.1 Prevention Formulation. Prevention methods aim to mitigate the impact of poisoning attacks. One effective defence involves a form of robust optimisation [81]. Formally, let $\mathcal{R}$ denote the set of observed ratings in the user-item interaction matrix. The objective function is then formulated as

$$\mathcal{E} = \mathcal{L} + \lambda * \mathcal{R}. \tag{19}$$

In this formulation, $\mathcal{L}$ denotes the original loss function, which measures the prediction error between the recommended ratings and the actual ratings. The term $\lambda * \mathcal{R}$ introduces a regularisation component into the objective function, promoting the generation of ratings that align with the observed ratings in the training data [125]. The coefficient λ determines the balance between accuracy and the system's resilience against poisoning attacks.

5.2.2 Mitigating Challenges to Achieve Effective Robustness. Most prevention methods rely on a combination of techniques. For example, the issue of *openness* is handled through outlier detection, data sanitisation, and preprocessing techniques [106]. Through such methods, manipulated data can be identified and filtered out. As another example, *concept drift* is often handled by using a dynamic learning algorithm that adapts to evolving user behaviour [113]. This adaptability allows the system to distinguish between genuine changes and fake users to help preserve accurate recommendations. *Imbalanced data* is generally addressed via ensemble methods [96] that assign more weight to the minority classes and use an over- or undersampling technique to balance the dataset.

5.2.3 Related Work. In addition to methods for detecting attacks, there is a line of research that aims to make a recommender system more robust to poisoning attacks without explicitly trying to detect them. More precisely, these algorithms strive to address the vulnerabilities in recommender systems that poisoning attacks exploit. This section briefly highlights existing works in this direction.

In their initial study, Sandvig et al. [102] (**CM-36**) found that model-based recommender systems are more stable and robust than memory-based recommender systems. Building on this, they devised a robust recommender system based on association rule mining, which significantly improves upon neighbour- and model-based recommender systems. Their algorithm captures item relationships through co-occurrences in user profiles, leading to more precise recommendations and mitigating the impact of poisoning attacks. Another study by Mehta et al. [80] (**CM-37**) explores the use of statistical techniques, specifically robust M-estimators, for achieving robustness in recommender systems. They leverage robust M-estimators to propose a matrix factorisation algorithm that outperforms other latent semantic-based algorithms such as PLSA and SVD. Mehta et al. [81] (**CM-38**) present an attack-resistant algorithm for SVD-based collaborative filtering that integrates the efficiency of SVD-based detectors into the system. This algorithm improves both accuracy and robustness of the model. Inspired by graph-based models for capturing malicious profiles, GraphRfi [156] (**CM-39**) introduces a framework based on a **graph convolutional network (GCN)** for user representation learning. It incorporates dual-task learning with the aim of simultaneously maintaining robust recommendations while also detecting attacks.

Differential privacy [51] has been widely accepted as an excellent method for protecting machine learning systems against poisoning attacks. Hence, there have been several attempts to apply this idea to recommender systems as well, including by Wadhwa et al. [117] (**CM-40**). These studies demonstrate that differentially private recommender systems are more robust than other types of systems and that, in most cases, attack utility is reduced. Anelli et al. [16] (**CM-41**) devised the idea of using adversarial training to enhance the robustness of **visual recommender systems (VRSs)**, exploring adversarial attacks and defence strategies specifically tailored to VRSs. Wu et al. [125] (**CM-42**) presented the APT scheme, which also involves adversarial training. This time, fake profiles are simulated, and the agents learn to generate fake users with minimal influence on the target recommender system's empirical risk. In turn, using both real and generated data strengthens the recommender system's robustness through dynamic training. The PORE framework, introduced by Jia et al. [66] (**CM-43**), builds recommender systems with a proven robustness against untargeted data poisoning attacks. PORE can convert any existing recommender system into a resilient one by limiting fake user ratings, incorporating knowledge from the recommender system algorithm, and establishing guarantees against targeted data poisoning attacks.

In summary, early studies on preventing poisoning attacks primarily focused on traditional analysis techniques, such as statistical methods and SVD-based methods. Later approaches have witnessed more elaborate foundations, such as GCNs and notions like differential privacy. Moreover, adversarial learning has proven to provide an excellent opportunity to obtain better representations of user profiles, thereby significantly improving the robustness of the systems.

5.3 Which Countermeasures Are Effective against Which Attacks?

Following the idea of attack visualisations [45], we developed a mapping of which countermeasures are effective against which attacks. The maps capture the attack traits and link them to the countermeasures as well as the strategies and capacities of an adversary. The mapping results are

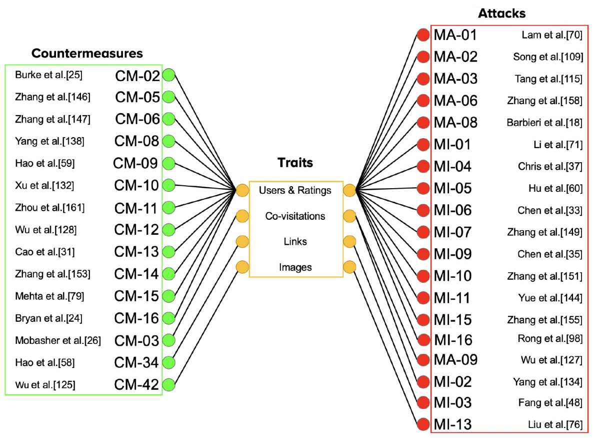

Fig. 6. Effective countermeasures against poisoning attacks.

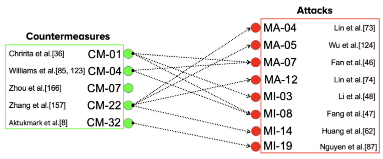

Fig. 7. Poisoning attacks that are resilient against certain countermeasures.

shown in Figure 6. Here, not surprisingly, we observe a predominant focus on the traits related to users and ratings, which highlights that many of the aforementioned strategies can actually be expected to achieve some level of robustness against a large number of attacks.

5.4 Which Countermeasures Are Weak against Which Attacks?

Most attacks do not only have the goal of promoting or demoting a set of items. Rather, they also strive to disguise themselves from countermeasures. While we have already considered this aspect in our review of poisoning attacks (see the "unnoticeable" column in Table 2), this mapping links these attacks to particular countermeasures, as shown in Figure 7. Here, weak countermeasures are listed in the left column, while the right column contains attacks that are resilient against certain countermeasures. In general, there are only a small number of countermeasures that are weak against specific attacks. It is also worth noting that each of the found dependencies constitutes an opportunity for research on how to avoid the corresponding weakness.

5.5 Summary

Finally, we provide an overview of the evolution of countermeasures against poisoning attacks in recommender systems. Figure 8 shows a timeline that captures the number of related studies in recent years. From this figure, we can see that countermeasure research received solid attention around 2019, whereas only a few studies have been published very recently.

6 Research Gaps, Limitations, and Future Directions

In this section, we discuss current research gaps and outline potential directions for future work in the field of poisoning attacks designed for recommender systems.

6.1 Gaps Related to Poisoning Attacks

Focusing on the design of poisoning attacks, we note that there are several open problems.

— **Adversary's knowledge: from white-box and grey-box to black-box attacks.** Existing poisoning attacks often rely on knowledge of the interactions between users and items, as is the case with white-box and grey-box attacks. However, privacy and security concerns typically limit access to targeted recommender systems. Although black-box attacks have

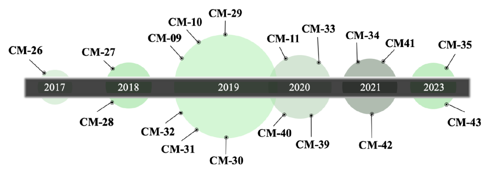

Fig. 8. The evolution of countermeasures.

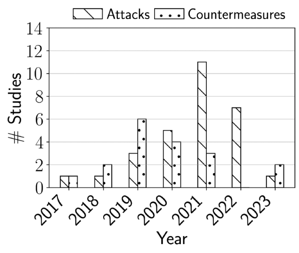

Fig. 9. Research trends for poisoning attacks and countermeasures for recommender systems.

gained attention, defining black-box attacks and finding countermeasures for them remains an open question requiring further research.

— **Attack approach: from specific to agnostic attacks.** Model-intrinsic attacks currently dominate the field. However, they are not particularly applicable to the real world. Future research could therefore focus on model-agnostic attacks to assess the robustness of emerging recommender systems.

— **Side-information fusion.** Most existing attacks follow the paradigm of injecting fake users and their interactions along with a small set of filler items into the training data (as discussed in Table 2). However, more recently, more and more recommender systems are beginning to also exploit side information to improve the quality of recommendations, such as domain hypergraphs [56], the social properties of users [143], and spatial information [112, 119]. Although side information can help, it also introduces additional complexities and potential security risks. By including side information, a recommender system might become more susceptible to poisoning attacks, because the attacker can manipulate the side information to manipulate the recommendations and mislead users. This emerging strategy deserves special attention, as it has the potential to undermine the reliability of recommender systems [139].

6.2 Gaps Related to Countermeasures

Next, we turn to the research gaps relating to countermeasures.

— **Pre-attack detection.** Most current countermeasures primarily focus on detecting poisoning attacks after they have already caused irreversible damage. Few approaches attempt to detect pre-attack behaviour, which involves identifying the malicious activities aimed at preparing an attack. Future research should not only look to identify the kinds of abnormal behaviour that precede an attack but also design mechanisms to monitor this type of behaviour in real-time. This would see a new category of countermeasures emerge in the form of a prediction method to sit alongside prevention and detection.

— **Going beyond the main traits.** As shown in Table 6, existing detection methods primarily focus on key indicators such as the ratings of filler and target items. However, supplementary information, such as a user's attributes, play an equally important role in preventing unforeseen damage to the attacked system. For example, if a new user is created with suspicious attributes, then the system should implement a mechanism to verify the user before incorporating their interactions. The same approach can be applied to the attributes of newly added items (filler items) to prevent manipulation based on these items. Therefore, we believe that incorporating additional traits into detection methods to identify and mitigate poisoning attacks represents a promising research direction.

— **Going beyond accuracy.** The pursuit of accuracy has traditionally been the primary goal of attack detection models. However, it has been recognised in recent years that focusing solely on accuracy is insufficient [58]. Ensuring fairness and explainability of the models has become equally important [5, 72]. Specifically, explainability plays a crucial role in developing countermeasures by understanding the creation of poisoning attacks and deriving preventive measures against similar attacks. Therefore, there is a need for research on explainability techniques to address vulnerabilities in recommender system models.

— **Overhead optimisation.** Our article primarily focuses on countering poison attacks in recommender systems. However, we acknowledge that the discussed applications may have broader implications. As a future direction, we propose optimising the overhead associated with feature extraction from historical data. This optimisation is crucial due to the significant performance impact on recommender systems. Real-time detection of fake users is essential for effective countermeasures, necessitating further emphasis on reducing overhead.

6.3 Limitations of the Work

Our limitation is the absence of a quantitative performance comparison between the reviewed poison attacks. Due to space constraints, we were unable to provide direct empirical evidence regarding the practical efficacy of these models. However, in future work, we plan to conduct a comprehensive quantitative performance comparison using real-world datasets and established evaluation metrics. By addressing this limitation in a future technical report, in conjunction with the current survey, our aim is to provide a more practical and comprehensive understanding of poison attacks in recommender systems and contribute to advancing this crucial research area.

6.4 Linking Poisoning Attacks and Countermeasures

Our analysis of published studies on poisoning attacks and countermeasures, as shown in Figure 9, reveals a concerning trend. While there has been a significant amount of research on novel types of poisoning attacks, there has not been much work in recent years on developing countermeasures. This indicates that state-of-the-art recommendation systems will have various vulnerabilities that can potentially be exploited and where there is no effective countermeasure to stop the attack. Therefore, a significant research gap exists that calls for further investigation on how to assess and enhance the robustness of recommender systems. This research is crucial for maintaining and partially restoring the trustworthiness and fairness of the underlying recommendation models.

6.5 Integrating General Vulnerabilities of Recommender Systems

In addition to the discussed poisoning attacks, recommender systems are susceptible to vulnerabilities arising from general security threats faced by information systems. However, these vulnerabilities have been largely overlooked and only informally addressed [54]. Consequently, there is an urgent need to systematically describe, analyse, and establish connections between these general vulnerabilities and attacks on recommender systems. As such, rigorous methodologies and design principles need to be developed, as such efforts are crucial for quantitatively assessing the impact of attacks and designing effective techniques to mitigate their associated risks.

6.6 Tools and Environments for Discovering Vulnerabilities

As is evident from Table 6, existing attack detection techniques do not cover all the vulnerabilities in today's recommender systems. Therefore, building a comprehensive countermeasure means combining different techniques. Tools that enable the seamless integration of various detection techniques would help in this regard. Additionally, practical research on novel attacks and countermeasures heavily relies on simulation environments for contemporary recommender system models, considering the limited access to production systems. Future work should prioritise the development of such simulation environments. These resources will serve as valuable testbeds for both practitioners and researchers, contributing to the advancement of the field as a whole.

7 Conclusion

In this survey, we presented a comprehensive overview of poisoning attacks for recommender systems along with countermeasures to detect and prevent them. We first distinguished poisoning attacks from similar concepts, such as adversarial attacks, before introducing a novel taxonomy for poisoning attacks. We formalised the dimensions of this taxonomy and linked a total of 31 attacks described in the literature to it. We complemented the discussion of attacks with a review of 43 countermeasures that have been proposed to detect or prevent poisoning attacks. Further insights have been provided by linking attacks and countermeasures, highlighting which countermeasures can be expected to be effective against which attacks, and which attacks can be expected to be resilient against which countermeasures. We concluded by highlighting research gaps and providing directions for future work. By providing a public repository that includes all reviewed papers along with the program codes and datasets released in the context of these studies, we have also provided researchers in the field a comprehensive starting point to address these open challenges.

Appendices

A Technical Aspects of Model-intrinsic Recommender Systems

This section provides a comprehensive exploration of various technical approaches employed in recommender systems, including recommender systems based on matrix factorisation, graphs, neighbourhoods, and deep learning, as well as federated recommender systems.

A.1 Matrix-factorisation-based Recommender Systems

In collaborative filtering, the objective is to decompose the user-item interaction matrix into separate matrices representing the latent factors of the users and items [68]. The user-item interaction matrix, denoted as $\mathcal{R}$, represents the interactions between users and items. In this matrix, r_{ui} denotes the specific interaction between user u and item i. The factorisation can be expressed as follows:

$$\mathcal{R} \approx U * V^T, \tag{20}$$

where U denotes the user latent factor matrix with a dimension of $m \times k$, where m represents the number of users and k represents the number of latent factors. V represents the item latent factor matrix with a dimension of $n \times k$, where n corresponds to the number of items. The predicted rating for user u and item i, denoted as $\tilde{r}_{ui}$, can be computed as the dot product of the respective latent factor vectors:

$$\tilde{r}_{ui} = U[i,:] * V[j,:]. \tag{21}$$

Matrix-factorisation-based recommender systems can be enhanced by including regularisation terms, bias terms, and other techniques to improve performance and address additional factors such as user/item biases and temporal dynamics [10]. Nevertheless, the fundamental principle remains centred around decomposing the rating matrix into low-rank user and item factor matrices, which means personalised recommendations can be generated [10, 68].

A.2 Graph-based Recommender Systems

In a graph-based recommender system, users, items, and their relationships are modelled using a graph structure [57] denoted as G = (V, E). Such graphs are constructed using the following procedure:

— The user-item relationship can be represented as an edge, denoted as e = (u, i), which indicates that user u has rated item i. Mathematically, this can be defined as:

$$e = (u, i) \in E, \text{if } r_{ui} \text{ is not null or zero.} \tag{22}$$

— Item-item relationships are also represented as edges, denoted as e = (i, j), which indicates the similarity between items i and j. Mathematically, this can be defined as:

$$e = (i, j) \in E, \text{if } sim(i, j) > \tau, \tag{23}$$

where $sim(i, j)$ represents a similarity measure between items i and j, while the threshold τ determines the minimum required similarity for an edge to be established.

Using the graph structure, recommendations can be generated by leveraging the relationships between users and items, as well as item-item relationships. The specific recommendation algorithms and techniques can vary, depending on the goals and characteristics of the recommender system [65].

A.3 Neighbourhood-based Recommender Systems

In neighbourhood-based recommender systems, the selection of neighbours holds significant importance. Neighbours are typically chosen based on their similarity to the target user or item [92]. Several similarity metrics, such as Pearson correlation coefficient or cosine similarity, can be used to quantify the similarity between users or items. After selecting the neighbourhood, the recommender system predicts the rating for a target user-item pair by considering the ratings of the neighbours. One common approach is weighted average rating prediction, where the predicted rating $\tilde{r}_{ui}$ for a target user u and item i can be calculated using the following formula:

$$\tilde{r}_{ui} = \frac{\sum (sim(u, v) * r_{ui})}{\sum sim(u, v)}, \tag{24}$$

where $sim(u, v)$ denotes the similarity between users u and v, r_{ui} represents the rating of user v for item i, and the summation is performed over the selected neighbourhood of user u [100].

A.4 Deep-learning-based Recommender Systems

Deep-learning-based recommender systems rely on neural network architectures to capture intricate patterns and representations from the data encompassing user-item interactions [86]. Suppose we examine a deep learning model characterised by the parameters θ. This model accepts user (z_u) and item (z_i) embeddings as its input and generates predictions for ratings or the probabilities of any interactions between the users and items [154]. Mathematically, the forward propagation in the deep learning model can be expressed as follows:

$$\tilde{\mathcal{R}} = f_\theta(z_u, z_i), \tag{25}$$

where the function f_θ represents the mapping function, which is parameterised by θ and is responsible for calculating the predicted rating or the probability of interaction between the user embedding (z_u) and the item embedding (z_i) [11].

A.5 Federated Recommender Systems

The main distinction between a federated recommender system and a traditional recommender system lies in the approach to data sharing [155]. Unlike a traditional recommender system, a federated recommender system does not share the complete rating matrix with any external entity [126]. Instead, users, items, and ratings are dispersed across various local data sources. The global model represents a comprehensive collection of knowledge, and recommendations are generated by combining insights from various local models. The collaborative learning process involves aggregating information from these local models:

— Local Training: This process involves training a machine learning model using a subset of the rating matrix $\mathcal{R}$ without the need to share the entire rating matrix with any other party.
— Global Aggregation: In this step, the global model parameters θ are obtained by aggregating the local model parameters $\{\theta_1, \theta_2, \ldots, \theta_k\}$ from individual data sources $\{L_1, L_2, \ldots, L_k\}$. The aggregation operation combines these parameters using techniques such as averaging or weighted aggregation. For instance, the global model can be calculated as $M = \frac{1}{N}\sum_{i=1}^{N} M_i$, where N represents the number of participating nodes and M_i corresponds to the local model at node i.

Federated recommender systems leverage a collaborative global model to generate recommendations by using aggregated knowledge from local data sources. This innovative approach to recommender systems shows great promise in tackling many challenges related to privacy and security [98].

B Evaluation Protocols for Countermeasures

The following section presents a comprehensive overview of the evaluation protocols, datasets, and domains used in various methods to counter poison attacks. Table 7 presents a summary of the evaluation protocols, while Table 8 lists the most commonly used datasets.

Table 7. Metrics Used in Countermeasures

Abbrv	Name	Abbrv	Name
RDMA	Rating Deviation from Mean Agreement	MSEP	Mean Similarity-based Expected Profit
ASM	Average Similarity Metric	Hit	Hit Rate Ratio
Spe	Specificity	CS	Classification Error
Sen	Sensitivity	DR	Detection Rate
FPR	False Positive Rate	FAR	False Alarm Rate
Pre	Precision	Hv-Score	Information Gain
Rec	Recall	MAS	Mean Absolute Shift
Acc	Accuracy	RMSS	Root Mean Square Shift
F1	F1-measure	MTP	Mean of Total Profit
F2	F2-measure–recall has twice weight of precision	PANT	Predicted Anomalous is not the Target Items
AUC	Area under the ROC Curve	MDD	Mean Detection Delay
MAE	Mean Absolute Error	Time	Timeliness
PS	Prediction Shift		

Table 8. Datasets Used in Countermeasures

Abbrv	Term
SYN	Synthetic datasets
ML	MovieLens
NF	Netflix
AMB	Amazon Book
AMM	Amazon Movie
AMP	Amazon Product
CI	Ciao DVD
EP	Epinions
EM	Eachmovie
BC	Book-Crossing
Trip	TripAdvisor
GOW	Gowalla
YE	Yelp

And Table 9 provides a summary of countermeasures, including their evaluation metrics, domains, and datasets adopted.

Evaluation Metrics. When evaluating poison attack detection methods in recommender systems, it is crucial to establish acceptable ranges for evaluation metrics. The ranges for metrics such as accuracy, precision, recall, and F1-score vary, depending on the context of poison attack detection [141]. Factors such as the recommender system's nature, the potential consequences of false positives, and the prevalence of poison attacks in the dataset impact these ranges. Simply achieving high accuracy may not suffice due to imbalanced datasets leading to misleading results [79]. Precision holds particular significance in poison attack detection, especially in scenarios where false positives can promote harmful or malicious content [59]. Although specific benchmarks for poison attack detection in recommender systems may not be explicitly defined, it is reasonable to consider models with performance scores around 0.8 to 0.9 as acceptable [113]. This range suggests the method correctly identifies poison attacks approximately 80% to 90% of the time. An excellent performance would be indicated by a score exceeding 0.9, indicating a robust defence

Table 9. Domain and Evaluation Comparison

Name	Authors	Year	RDMA	ASM	Spe	Sen	FPR	Pre	Rec	Acc	F1	F2	AUC	MAE	MSEP	Hit	PS	CS	DR	FAR	Hv-Score	MAS	RMSS	MTP	MDD	PANT	Time	Domain	Dataset
			Evaluation Metric																										
	Supervised																												
CM-01	Chirita et al. [37]	2005	✓	✓																								movie	ML
CM-02	Burke et al. [26]	2006						✓	✓																			movie	ML
CM-03	Mobasher et al. [27]	2006										✓	✓	✓			✓											movie	ML
CM-04	Williams et al. [85, 123]	2007						✓	✓							✓	✓											movie	ML
CM-05	Zhang et al. [146]	2012						✓	✓																			movie	ML
CM-06	Zhang et al. [147]	2014			✓	✓		✓																			✓	movie	ML
CM-07	Zhou et al. [166]	2016					✓	✓	✓																			movie	ML
CM-08	Yang et al. [138]	2016																✓	✓	✓								movie	ML
CM-09	Hao et al. [60]	2019						✓	✓																			movie, product	NF, ML, AMM
CM-10	Xu et al. [132]	2019						✓	✓		✓																	product	CI, EP
CM-11	Zhou et al. [161]	2020						✓	✓		✓		✓															movie	ML
	Semi-supervised																												
CM-12	Wu et al. [128]	2012						✓	✓		✓			✓														movie	NF, ML
CM-13	Cao et al. [32]	2013			✓	✓																						movie	ML
	Unsupervised																												
CM-14	Zhang et al. [153]	2006																	✓	✓								movie	ML
CM-15	Mehta et al. [79]	2007						✓		✓																		movie	ML
CM-16	Bryan et al. [25]	2008																			✓							movie	ML
CM-17	Meta et al. [82]	2009						✓	✓					✓														movie	ML
CM-18	Bhaumik et al. [21]	2011			✓	✓															✓							movie	ML
CM-19	Chung et al. [39]	2013																	✓	✓								movie	ML
CM-20	Bilge et al. [22]	2014						✓	✓		✓																	movie	ML
CM-21	Zhou et al. [162]	2014					✓												✓									movie	NF, ML
CM-22	Zhang et al. [157]	2015						✓	✓		✓																	product	AMM
CM-23	Zhou et al. [164]	2015			✓	✓	✓																					movie	ML, NF, EM
CM-24	Xia et al. [130]	2015																				✓	✓				✓	movie	ML
CM-25	Yang et al. [135]	2016																	✓	✓								movie	ML
CM-26	Yang et al. [136]	2017																	✓	✓								movie, product	ML, AMP
CM-27	Zhang et al. [145]	2018						✓	✓		✓																	movie, product	ML, NF, AMP
CM-28	Zhou et al. [165]	2018					✓												✓									movie	ML
CM-29	Cai et al. [28]	2019						✓	✓		✓																	synthetic, product	SYN, AMP
CM-30	Cai et al. [29]	2019						✓	✓		✓																	movie, product	NF, ML, AMP
CM-31	Cai et al. [30]	2019						✓	✓		✓																	book	BC, AMB
CM-32	Aktukmak et al. [9]	2019											✓												✓			movie	ML
CM-33	Yang et al. [137]	2020																	✓	✓						✓		movie, book, hotel	ML, AMB, LT, Trip
CM-34	Hao et al. [59]	2021						✓	✓																			movie, product	NF, AMM
CM-35	You et al. [141]	2023							✓	✓	✓																	POI, Location	GOW, YE

against poison attacks [96]. In future research, we aim to conduct comparative evaluations and establish concrete acceptable ranges for poison attack detection metrics in recommender systems. These findings will be presented in a technical report.

Domains of Application. Countermeasures for poison attacks in recommender systems reveal a primary focus on the movie domain, and this emphasis highlights the significance of safeguarding movie recommender systems against manipulation [9, 59, 161]. However, researchers have also recognised the broader implications of poison attacks and have developed countermeasures that address multiple domains, including products [59], books [30], hotels [137], and synthetic data [28]. Moreover, the inclusion of POI and location [141] as domains of interest reflects the concern for protecting location-based recommender systems. By covering these diverse domains, the reviewed countermeasures aim to enhance the security and reliability of recommender systems, preserving user trust and satisfaction in their recommendations across a wide range of applications.

References

[1] IndustryARC. 2024. Recommendation Engine Market - Forecast(2024 - 2030). Retrieved 3 August 2024 from https://www.industryarc.com/Research/Recommendation-Engine-Market-Research-500995

[2] BBC News. 2001. Sony admits using fake reviewer. Retrieved 3 August 2024 from http://news.bbc.co.uk/2/hi/entertainment/1368666.stm

[3] Penta Security. 2021. Top 5 AI-powered cyber threats & how to prevent them. Retrieved from 3 August 2024 https://www.pentasecurity.com/blog/top-5-ai-powered-cyber-threats-how-to-prevent-them/

[4] Thanh Tam Nguyen. 2024. Github - Awesome Recsys Poisoning. Retrieved 3 August 2024 from https://github.com/tamlhp/awesome-recsys-poisoning

[5] Behnoush Abdollahi and Olfa Nasraoui. 2018. Transparency in fair machine learning: The case of explainable recommender systems. In *Human and Machine Learning*. Springer, 21–35.

[6] P. H. Aditya, Indra Budi, and Qorib Munajat. 2016. A comparative analysis of memory-based and model-based collaborative filtering on the implementation of recommender system for E-commerce. In *ICACSIS*. 303–308.

[7] Alankrita Aggarwal, Mamta Mittal, and Gopi Battineni. 2021. Generative adversarial network: An overview of theory and applications. *Int. J. Inf. Manag. Data Insights* 1, 1 (2021), 100004.

[8] C. C. Aggarwal. 2016. Recommender systems (Vol. 1). Cham: Springer International Publishing. https://link.springer.com/book/10.1007/978-3-319-29659-3

[9] Mehmet Aktukmak, Yasin Yilmaz, and Ismail Uysal. 2019. Quick and accurate attack detection in recommender systems through user attributes. In *RecSys*. 348–352.

[10] Bushra Alhijawi and Yousef Kilani. 2020. The recommender system: A survey. *Int. J. Advan. Intell. Parad.* 15, 3 (2020), 229–251.

[11] Zafar Ali, Pavlos Kefalas, Khan Muhammad, Bahadar Ali, and Muhammad Imran. 2020. Deep learning in citation recommendation models survey. *Expert Syst. Applic.* 162 (2020), 113790.

[12] Zafar Ali, Shah Khusro, and Irfan Ullah. 2016. A hybrid book recommender system based on table of contents (toc) and association rule mining. In *INFOS*. 68–74.

[13] Zafar Ali, Irfan Ullah, Amin Khan, Asim Ullah Jan, and Khan Muhammad. 2021. An overview and evaluation of citation recommendation models. *Scientometrics* 126 (2021), 4083–4119.

[14] E. Aliwa, O. Rana, C. Perera, and P. Burnap. 2021. Cyberattacks and countermeasures for in-vehicle networks. *ACM Computing Surveys (CSUR)*, 54, 1(2021), 1–37.

[15] Yair Amir, Brian Coan, Jonathan Kirsch, and John Lane. 2008. Byzantine replication under attack. In *DSN*. 197–206.

[16] Vito Walter Anelli, Yashar Deldjoo, Tommaso Di Noia, Daniele Malitesta, and Felice Antonio Merra. 2021. A study of defensive methods to protect visual recommendation against adversarial manipulation of images. In *SIGIR*.

[17] Khalid Anwar, Jamshed Siddiqui, and Shahab Saquib Sohail. 2020. Machine learning-based book recommender system: A survey and new perspectives. *Int. J. Intell. Inf. Datab. Syst.* 13, 2-4 (2020), 231–248.

[18] Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, and Jaehoon Amir Safavi. 2017. Mitigating poisoning attacks on machine learning models: A data provenance based approach. In *AISec*. 103–110.

[19] Julio Barbieri, Leandro G. M. Alvim, Filipe Braida, and Geraldo Zimbrão. 2021. Simulating real profiles for shilling attacks: A generative approach. *Knowl-based Syst.* 230 (2021), 107390.

[20] Adam Barth, Collin Jackson, and John C. Mitchell. 2008. Robust defenses for cross-site request forgery. In *CCS*. 75–88.

[21] Runa Bhaumik, Bamshad Mobasher, and Robin Burke. 2011. A clustering approach to unsupervised attack detection in collaborative recommender systems. In *ICDATA*. 1.

[22] A. Bilge, Z. Ozdemir, and H. Polat. 2014. A novel shilling attack detection method. *Procedia Computer Science*, 31 (2014), 165–174.

[23] P. Branco, L. Torgo, and R. P. Ribeiro. 2016. A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR)*, 49, 2 (2016), 1–50.

[24] Kenneth Bryan and Pádraig Cunningham. 2006. Bottom-up biclustering of expression data. In *CIBCB*. 1–8.

[25] Kenneth Bryan, Michael O'Mahony, and Pádraig Cunningham. 2008. Unsupervised retrieval of attack profiles in collaborative recommender systems. In *RecSys*. 155–162.

[26] Robin Burke, Bamshad Mobasher, Chad Williams, and Runa Bhaumik. 2006. Classification features for attack detection in collaborative recommender systems. In *KDD*. 542–547.

[27] Robin Burke, Bamshad Mobasher, Chad Williams, and Runa Bhaumik. 2006. Detecting profile injection attacks in collaborative recommender systems. In *CEC/EEE)*. 23–23.

[28] H. Cai and F. Zhang. 2019. BS-SC: An unsupervised approach for detecting shilling profiles in collaborative recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, 33, 4 (2019), 1375–1388.

[29] Hongyun Cai and Fuzhi Zhang. 2019. Detecting shilling attacks in recommender systems based on analysis of user rating behavior. *Knowl-based Syst.* 177 (2019), 22–43.

[30] Yuanfeng Cai and Dan Zhu. 2019. Trustworthy and profit: A new value-based neighbor selection method in recommender systems under shilling attacks. *Decis. Supp. Syst.* 124 (2019), 113112.

[31] Zhipeng Cai, Zuobin Xiong, Honghui Xu, Peng Wang, Wei Li, and Yi Pan. 2021. Generative adversarial networks: A survey toward private and secure applications. *Comput. Surv.* 54, 6 (2021), 1–38.

[32] Jie Cao, Zhiang Wu, Bo Mao, and Yanchun Zhang. 2013. Shilling attack detection utilizing semi-supervised learning method for collaborative recommender system. *World Wide Web* 16, 5-6 (2013), 729–748.

[33] Henry Chacon, Samuel Silva, and Paul Rad. 2019. Deep learning poison data attack detection. In *ICTAI*. 971–978.

[34] Huiyuan Chen and Jing Li. 2019. Data poisoning attacks on cross-domain recommendation. In *CIKM*. 2177–2180.

[35] Jingfan Chen, Wenqi Fan, Guanghui Zhu, Xiangyu Zhao, Chunfeng Yuan, Qing Li, and Yihua Huang. 2022. Knowledge-enhanced black-box attacks for recommendations. In *KDD*. 108–117.

[36] Liang Chen, Yangjun Xu, Fenfang Xie, Min Huang, and Zibin Zheng. 2021. Data poisoning attacks on neighborhood-based recommender systems. *Trans. Emerg. Telecommun. Technol.* 32, 6 (2021), e3872.

[37] Paul-Alexandru Chirita, Wolfgang Nejdl, and Cristian Zamfir. 2005. Preventing shilling attacks in online recommender systems. In *WIDM*. 67–74.

[38] Konstantina Christakopoulou and Arindam Banerjee. 2019. Adversarial attacks on an oblivious recommender. In *RecSys*. 322–330.

[39] Chen-Yao Chung, Ping-Yu Hsu, and Shih-Hsiang Huang. 2013. βP: A novel approach to filter out malicious rating profiles from recommender systems. *Decis. Supp. Syst.* 55, 1 (2013), 314–325.

[40] Bin Dai, Shilin Ding, and Grace Wahba. 2013. Multivariate Bernoulli distribution. *Bernoulli* 19, 4 (2013), 1465–1483.

[41] D. Das, L. Sahoo, and S. Datta. 2017. A survey on recommendation system. *International Journal of Computer Applications* 160, 7 (2017), 6–10.

[42] Nicola De Bellis. 2009. *Bibliometrics and Citation Analysis: From the Science Citation Index to Cybermetrics*. SP.

[43] Yashar Deldjoo, Tommaso Di Noia, and Felice Antonio Merra. 2021. A survey on adversarial recommender systems: From attack/defense strategies to generative adversarial networks. *Comput. Surv.* 54, 2 (2021), 1–38.

[44] Zhou Dengwen. 2010. An edge-directed bicubic interpolation algorithm. In *CISP*. 1186–1189.

[45] M. Evangelopoulou and C. W. Johnson. 2014. Attack visualisation for cyber-security situation awareness.

[46] Jiaxin Fan, Qi Yan, Mohan Li, Guanqun Qu, and Yang Xiao. 2022. A survey on data poisoning attacks and defenses. In *DSC*. IEEE, 48–55.

[47] Wenqi Fan, Tyler Derr, Xiangyu Zhao, Yao Ma, Hui Liu, Jianping Wang, Jiliang Tang, and Qing Li. 2021. Attacking black-box recommendations via copying cross-domain user profiles. In *ICDE*. 1583–1594.

[48] Minghong Fang, Neil Zhenqiang Gong, and Jia Liu. 2020. Influence function based data poisoning attacks to top-n recommender systems. In *WWW*. 3019–3025.

[49] Minghong Fang, Guolei Yang, Neil Zhenqiang Gong, and Jia Liu. 2018. Poisoning attacks to graph-based recommender systems. In *ACSAC*. 381–392.

[50] S. Fletcher and M. Z. Islam. 2019. Decision tree classification with differential privacy: A survey. *ACM Computing Surveys (CSUR)*, 52, 4 (2019), 1–33.

[51] Arik Friedman and Assaf Schuster. 2010. Data mining with differential privacy. In *KDD*. 493–502.

[52] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. *Comput. Surv.* 46, 4 (2014), 1–37.

[53] Joseph Gardiner and Shishir Nagaraja. 2016. On the security of machine learning in malware C&C detection: A survey. *Comput. Surv.* 49, 3 (2016), 1–39.

[54] Seyed Mohammad Ghaffarian and Hamid Reza Shahriari. 2017. Software vulnerability analysis and discovery using machine-learning and data-mining techniques: A survey. *ACM Comput. Surv.* 50, 4 (2017), 1–36.

[55] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

[56] Lei Guo, Li Tang, Tong Chen, Lei Zhu, Quoc Viet Hung Nguyen, and Hongzhi Yin. 2021. DA-GCN: A domain-aware attentive graph convolution network for shared-account cross-domain sequential recommendation. *arXiv preprint arXiv:2105.03300* (2021).

[57] Q. Guo, F. Zhuang, C. Qin, H. Zhu, X. Xie, H. Xiong, and Q. He. 2020. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering* 34, 8 (2020), 3549–3568.

[58] Jungkyu Han and Hayato Yamana. 2017. A survey on recommendation methods beyond accuracy. *IEICE Trans. Inf. Syst.* 100, 12 (2017), 2931–2944.

[59] Yaojun Hao and Fuzhi Zhang. 2021. An unsupervised detection method for shilling attacks based on deep learning and community detection. *Soft Comput.* 25, 1 (2021), 477–494.

[60] Y. Hao, F. Zhang, J. Wang, Q. Zhao, and J. Cao. 2019. Detecting shilling attacks with automatic features from multiple views. *Security and Communication Networks* 2019, 1 (2019), 6523183.

[61] Rui Hu, Yuanxiong Guo, Miao Pan, and Yanmin Gong. 2019. Targeted poisoning attacks on social recommender systems. In *GLOBECOM*. 1–6.

[62] Y. Hu, Y. Koren, and C. Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 263–272.

[63] Hai Huang, Jiaming Mu, Neil Zhenqiang Gong, Qi Li, Bin Liu, and Mingwei Xu. 2021. Data poisoning attacks to deep learning based recommender systems. *arXiv preprint arXiv:2101.02644* (2021).

[64] Keman Huang, Michael Siegel, and Stuart Madnick. 2018. Systematically understanding the cyber attack business: A survey. *Comput. Surv.* 51, 4 (2018), 1–36.

[65] Zan Huang, Wingyan Chung, Thian-Huat Ong, and Hsinchun Chen. 2002. A graph-based recommender system for digital library. In *JCDL*. 65–73.

[66] Jinyuan Jia, Yupei Liu, Yuepeng Hu, and Neil Zhenqiang Gong. 2023. PORE: Provably robust recommender systems against data poisoning attacks. *arXiv preprint arXiv:2303.14601* (2023).

[67] Wazir Zada Khan, Mohammed Y. Aalsalem, Mohammed Naufal Bin Mohammed Saad, and Yang Xiang. 2013. Detection and mitigation of node replication attacks in wireless sensor networks: A survey. *Int. J. Distrib. Sensor Netw.* 9, 5 (2013), 149023.

[68] Y. Koren, R. Bell, and C. Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[69] S. K. T. Lam, D. Frankowski, and J. Riedl. 2006. Do you trust your recommendations? An exploration of security and privacy issues in recommender systems. In *International Conference on Emerging Trends in Information and Communication Security*. Berlin, Heidelberg: Springer Berlin Heidelberg, 14–29.

[70] Shyong K. Lam and John Riedl. 2004. Shilling recommender systems for fun and profit. In *WWW*. 393–402.

[71] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. 2016. Data poisoning attacks on factorization-based collaborative filtering. In *NIPS*. 1885–1893.

[72] Yunqi Li, Yingqiang Ge, and Yongfeng Zhang. 2021. Tutorial on fairness of machine learning in recommender systems. In *SIGIR*. 2654–2657.

[73] Chen Lin, Si Chen, Hui Li, Yanghua Xiao, Lianyun Li, and Qian Yang. 2020. Attacking recommender systems with augmented user profiles. In *CIKM*. 855–864.

[74] C. Lin, S. Chen, M. Zeng, S. Zhang, M. Gao, and H. Li. 2022. Shilling black-box recommender systems by learning to generate fake user profiles. *IEEE Transactions on Neural Networks and Learning Systems* 35, 1 (2022), 1305–1319.

[75] Yanyan Lit, Sara Kim, and Eric Sy. 2021. A survey on Amazon Alexa attack surfaces. In *CCNC*. 1–7.

[76] Zhuoran Liu and Martha Larson. 2021. Adversarial item promotion: Vulnerabilities at the core of Top-N recommenders that use images to address cold start. In *WWW*. 3590–3602.

[77] Wim Maes, Thomas Heyman, Lieven Desmet, and Wouter Joosen. 2009. Browser protection against cross-site request forgery. In *SecuCode*. 3–10.

[78] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2021. A survey on bias and fairness in machine learning. *Comput. Surv.* 54, 6 (2021), 1–35.

[79] Bhaskar Mehta, Thomas Hofmann, and Peter Fankhauser. 2007. Lies and propaganda: Detecting spam users in collaborative filtering. In *IUI*. 14–21.

[80] Bhaskar Mehta, Thomas Hofmann, and Wolfgang Nejdl. 2007. Robust collaborative filtering. In *RecSys*. 49–56.

[81] Bhaskar Mehta and Wolfgang Nejdl. 2008. Attack resistant collaborative filtering. In *SIGIR*. 75–82.

[82] Bhaskar Mehta and Wolfgang Nejdl. 2009. Unsupervised strategies for shilling detection and robust collaborative filtering. *User Model. User-adapt. Interact.* 19, 1 (2009), 65–97.

[83] Joao Mendes-Moreira, Carlos Soares, Alípio Mário Jorge, and Jorge Freire De Sousa. 2012. Ensemble approaches for regression: A survey. *Comput. Surv.* 45, 1 (2012), 1–40.

[84] Wei Meng, Xinyu Xing, Anmol Sheth, Udi Weinsberg, and Wenke Lee. 2014. Your online interests: Pwned! a pollution attack against targeted advertising. In *SIGSAC*. 129–140.

[85] Bamshad Mobasher, Robin Burke, Runa Bhaumik, and Chad Williams. 2007. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *Trans. Internet Technol.* 7, 4 (2007), 23–es.

[86] Ruihui Mu. 2018. A survey of recommender systems based on deep learning. *IEEE Access* 6 (2018), 69009–69022.

[87] Toan Nguyen Thanh, Nguyen Duc Khang Quach, Thanh Tam Nguyen, Thanh Trung Huynh, Viet Hung Vu, Phi Le Nguyen, Jun Jo, and Quoc Viet Hung Nguyen. 2023. Poisoning GNN-based recommender systems with generative surrogate-based attacks. *ACM Trans. Inf. Syst.* 41, 3 (2023), 1–24.

[88] Michael P. O'Mahony, Neil J. Hurley, and Guenole Silvestre. 2002. Promoting recommendations: An attack on collaborative filtering. In *DEXA*. 494–503.

[89] Claire B. Owen. 2008. *Parameter Estimation for the Beta Distribution.* Brigham Young University.

[90] Sindhu Padakandla. 2021. A survey of reinforcement learning algorithms for dynamically varying environments. *Comput. Surv.* 54, 6 (2021), 1–25.

[91] Shubham Pateria, Budhitama Subagdja, Ah-hwee Tan, and Chai Quek. 2021. Hierarchical reinforcement learning: A comprehensive survey. *Comput. Surv.* 54, 5 (2021), 1–35.

[92] K. Periyasamy, J. Jaiganesh, K. Ponnambalam, J. Rajasekar, and K. Arputharaj. 2017. Analysis and performance evaluation of cosine neighbourhood recommender system. *International Arab Journal of Information Technology (IAJIT)*, 14, 5 (2017).

[93] Nikolaos Pitropakis, Emmanouil Panaousis, Thanassis Giannetsos, Eleftherios Anastasiadis, and George Loukas. 2019. A taxonomy and survey of attacks against machine learning. *Comput. Sci. Rev.* 34 (2019), 100199.

[94] Nikolaos Polatidis, Elias Pimenidis, Michalis Pavlidis, and Haralambos Mouratidis. 2017. Recommender systems meeting security: From product recommendation to cyber-attack prediction. In *EANN*. 508–519.

[95] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M. L. Shyu, S. C. Chen, and S. S. Iyengar. 2018. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51, 5 (2018), 1–36.

[96] Fatemeh Rezaimehr and Chitra Dadkhah. 2021. A survey of attack detection approaches in collaborative filtering recommender systems. *Artif. Intell. Rev.* 54, 3 (2021), 2011–2066.

[97] F. Ricci, L. Rokach, and B. Shapira. 2010. Introduction to recommender systems handbook. In *Recommender Systems Handbook*. Boston, MA: springer US, 1–35.

[98] Dazhong Rong, Qinming He, and Jianhai Chen. 2022. Poisoning deep learning based recommender model in federated learning scenarios. *arXiv preprint arXiv:2204.13594* (2022).

[99] Ishai Rosenberg, Asaf Shabtai, Yuval Elovici, and Lior Rokach. 2021. Adversarial machine learning attacks and defense methods in the cyber security domain. *Comput. Surv.* 54, 5 (2021), 1–36.

[100] Atisha Sachan and Vineet Richariya. 2013. A survey on recommender systems based on collaborative filtering technique. *Int. J. Inf. Educ. Technol.* 2, 2 (2013), 8–14.

[101] Ruslan Salakhutdinov and Andriy Mnih. 2008. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *ICML*. 880–887.

[102] Jeff J. Sandvig, Bamshad Mobasher, and Robin Burke. 2007. Robustness of collaborative recommendation based on association rule mining. In *RecSys*. 105–112.

[103] Philip Sedgwick. 2012. Pearson's correlation coefficient. *Brit. Med. J.* 345 (2012).

[104] G. Shani and A. Gunawardana. 2011. Evaluating recommendation systems. In *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. Kantor, (Eds.). Springer, Boston, MA. https://doi.org/10.1007/978-0-387-85820-3_8

[105] Yue Shi, Martha Larson, and Alan Hanjalic. 2014. Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *Comput. Surv.* 47, 1 (2014), 1–45.

[106] Mingdan Si and Qingshan Li. 2020. Shilling attacks against collaborative recommender systems: A review. *Artif. Intell. Rev.* 53, 1 (2020), 291–319.

[107] J. Sidhu, R. Sakhuja, and D. Zhou. 2016. Attacks on eBay. *Lassonde School of Engineering*. https://www.eecs.yorku.ca/course_archive/2015-16/W/3482/Team12_eBayHacks.pdf

[108] Salima Smiti and Makram Soui. 2020. Bankruptcy prediction using deep learning approach based on borderline SMOTE. *Inf. Syst. Front.* 22, 5 (2020), 1067–1083.

[109] Junshuai Song, Zhao Li, Zehong Hu, Yucheng Wu, Zhenpeng Li, Jian Li, and Jun Gao. 2020. PoisonRec: An adaptive data poisoning framework for attacking black-box recommender systems. In *ICDE*. 157–168.

[110] F. Strub, R. Gaudel, and J. Mary. 2016. Hybrid recommender system based on autoencoders. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Syst.*

[111] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. 2019. One pixel attack for fooling deep neural networks. *IEEE Trans. Evolut. Comput.* 23, 5 (2019), 828–841.

[112] Ke Sun, Tieyun Qian, Tong Chen, Yile Liang, Quoc Viet Hung Nguyen, and Hongzhi Yin. 2020. Where to go next: Modeling long-and short-term user preferences for point-of-interest recommendation. In *AAAI*. 214–221.

[113] Agnideven Palanisamy Sundar, Feng Li, Xukai Zou, Tianchong Gao, and Evan D. Russomanno. 2020. Understanding shilling attacks and their detection traits: A comprehensive survey. *IEEE Access* 8 (2020), 171703–171715.

[114] Elham Tabassi, Kevin Burns, Michael Hadjimichael, Andres Molina-Markham, and Julian Sexton. 2019. A taxonomy and terminology of adversarial machine learning. Technical Report. NIST.

[115] Jiaxi Tang, Hongyi Wen, and Ke Wang. 2020. Revisiting adversarially learned injection attacks against recommender systems. In *RecSys*. 318–327.

[116] Shirin Tavara. 2019. Parallel computing of support vector machines: A survey. *Comput. Surv.* 51, 6 (2019), 1–38.

[117] Soumya Wadhwa, Saurabh Agrawal, Harsh Chaudhari, Deepthi Sharma, and Kannan Achan. 2020. Data poisoning attacks against differentially private recommender systems. In *SIGIR*. 1617–1620.

[118] J. Wang and Q. Tang. 2015. Recommender systems and their security concerns. Technical Report. University of Luxembourg.

[119] Qinyong Wang, Hongzhi Yin, Tong Chen, Zi Huang, Hao Wang, Yanchang Zhao, and Nguyen Quoc Viet Hung. 2020. Next point-of-interest recommendation on resource-constrained mobile devices. In *WWW*. 906–916.

[120] Xianmin Wang, Jing Li, Xiaohui Kuang, Yu-an Tan, and Jin Li. 2019. The security of machine learning in an adversarial setting: A survey. *J. Parallel Distrib. Comput.* 130 (2019), 12–23.

[121] Youquan Wang, Liqiang Qian, Fanzhang Li, and Lu Zhang. 2018. A comparative study on shilling detection methods for trustworthy recommendations. *J. Syst. Sci. Syst. Eng.* 27, 4 (2018), 458–478.

[122] Youquan Wang, Lu Zhang, Haicheng Tao, Zhiang Wu, and Jie Cao. 2015. A comparative study of shilling attack detectors for recommender systems. In *ICSSSM*. 1–6.

[123] Chad A. Williams, Bamshad Mobasher, and Robin Burke. 2007. Defending recommender systems: Detection of profile injection attacks. *Serv. Orient. Comput. Applic.* 1, 3 (2007), 157–170.

[124] Chenwang Wu, Defu Lian, Yong Ge, Zhihao Zhu, and Enhong Chen. 2021. Triple adversarial learning for influence based poisoning attack in recommender systems. In *KDD*. 1830–1840.

[125] Chenwang Wu, Defu Lian, Yong Ge, Zhihao Zhu, Enhong Chen, and Senchao Yuan. 2021. Fight fire with fire: Towards robust recommender systems via adversarial poisoning training. In *SIGIR*. 1074–1083.

[126] Chuhan Wu, Fangzhao Wu, Tao Qi, Yongfeng Huang, and Xing Xie. 2022. FedAttack: Effective and covert poisoning attack on federated recommendation via hard sampling. *arXiv preprint arXiv:2202.04975* (2022).

[127] F. Wu, M. Gao, J. Yu, Z. Wang, K. Liu, and X. Wang. 2021. Ready for emerging threats to recommender systems? A graph convolution-based generative shilling attack. *Information Sciences* 578 (2021), 683–701.

[128] Zhiang Wu, Junjie Wu, Jie Cao, and Dacheng Tao. 2012. HySAD: A semi-supervised hybrid shilling attack detector for trustworthy product recommendation. In *KDD*. 985–993.

[129] Z. W. Wu, C. T. Chen, and S. H. Huang. 2022. Poisoning attacks against knowledge graph-based recommendation systems using deep reinforcement learning. *Neural Computing and Applications*. 1–19.

[130] H. Xia, B. Fang, M. Gao, H. Ma, Y. Tang, and J. Wen. 2015. A novel item anomaly detection approach against shilling attacks in collaborative recommendation systems using the dynamic time interval segmentation technique. *Information Sciences* 306 (2015), 150–165.

[131] Xu Ximeng, Yang Rennong, and Fu Ying. 2018. Situation assessment for air combat based on novel semi-supervised naive Bayes. *J. Syst. Eng. Electron.* 29, 4 (2018), 768–779.

[132] Yishu Xu and Fuzhi Zhang. 2019. Detecting shilling attacks in social recommender systems based on time series analysis and trust features. *Knowl-based Syst.* 178 (2019), 25–47.

[133] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. 2017. Deep matrix factorization models for recommender systems. In *IJCAI*. 3203–3209.

[134] Guolei Yang, Neil Zhenqiang Gong, and Ying Cai. 2017. Fake co-visitation injection attacks to recommender systems. In *NDSS*.

[135] Zhihai Yang, Zhongmin Cai, and Xiaohong Guan. 2016. Estimating user behavior toward detecting anomalous ratings in rating systems. *Knowl-based Syst.* 111 (2016), 144–158.

[136] Zhihai Yang, Zhongmin Cai, and Yuan Yang. 2017. Spotting anomalous ratings for rating systems by analyzing target users and items. *Neurocomputing* 240 (2017), 25–46.

[137] Zhihai Yang, Qindong Sun, Yaling Zhang, and Wei Wang. 2020. Identification of malicious injection attacks in dense rating and co-visitation behaviors. *IEEE Trans. Inf. Forens. Secur.* 16 (2020), 537–552.

[138] Zhihai Yang, Lin Xu, Zhongmin Cai, and Zongben Xu. 2016. Re-scale AdaBoost for attack detection in collaborative filtering recommender systems. *Knowl-based Syst.* 100 (2016), 74–88.

[139] Dan Ye, Bing Yang, and Tian-Yu Zhang. 2021. Optimal stealthy linear attack on remote state estimation with side information. *IEEE Syst. J.* 16, 1 (2021), 1499–1507.

[140] Jingwei Yi, Fangzhao Wu, Bin Zhu, Yang Yu, Chao Zhang, Guangzhong Sun, and Xing Xie. 2022. UA-FedRec: Untargeted attack on federated news recommendation. *arXiv preprint arXiv:2202.06701* (2022).

[141] Xiaoyu You, Chi Li, Daizong Ding, Mi Zhang, Fuli Feng, Xudong Pan, and Min Yang. 2023. Anti-FakeU: Defending shilling attacks on graph neural network based recommender model. In *WWW*. 938–948.

[142] C. Yu, J. Liu, S. Nemati and G. Yin. 2021. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55, 1 (2021), 1–36.

[143] Junliang Yu, Hongzhi Yin, Min Gao, Xin Xia, Xiangliang Zhang, and Nguyen Quoc Viet Hung. 2021. Socially-aware self-supervised tri-training for recommendation. *arXiv preprint arXiv:2106.03569* (2021).

[144] Zhenrui Yue, Zhankui He, Huimin Zeng, and Julian McAuley. 2021. Black-box attacks on sequential recommenders via data-free model extraction. In *RecSys*. 44–54.

[145] Fuzhi Zhang, Zening Zhang, Peng Zhang, and Shilei Wang. 2018. UD-HMM: An unsupervised method for shilling attack detection based on hidden Markov model and hierarchical clustering. *Knowl-based Syst.* 148 (2018), 146–166.

[146] Fuzhi Zhang and Quanqiang Zhou. 2012. A meta-learning-based approach for detecting profile injection attacks in collaborative recommender systems. *J. Comput.* 7, 1 (2012), 226–234.

[147] Fuzhi Zhang and Quanqiang Zhou. 2014. HHT–SVM: An online method for detecting profile injection attacks in collaborative recommender systems. *Knowl-based Syst.* 65 (2014), 96–105.

[148] Guijuan Zhang, Yang Liu, and Xiaoning Jin. 2020. A survey of autoencoder-based recommender systems. *Front. Comput. Sci.* 14, 2 (2020), 430–450.

[149] Hengtong Zhang, Yaliang Li, Bolin Ding, and Jing Gao. 2020. Practical data poisoning attack against next-item recommendation. In *WWW*. 2458–2464.

[150] H. Zhang, Y. Li, B. Ding, and J. Gao. 2022. LOKI: a practical data poisoning attack framework against next item recommendations. *IEEE Transactions on Knowledge and Data Engineering* 35, 5 (2022), 5047–5059.

[151] Hengtong Zhang, Changxin Tian, Yaliang Li, Lu Su, Nan Yang, Wayne Xin Zhao, and Jing Gao. 2021. Data poisoning attack against recommender system using incomplete and perturbed data. In *KDD*. 2154–2164.

[152] Jing Zhang, Jie Tang, Cong Ma, Hanghang Tong, Yu Jing, and Juanzi Li. 2015. Panther: Fast top-k similarity search on large networks. In *KDD*. 1445–1454.

[153] Sheng Zhang, Amit Chakrabarti, James Ford, and Fillia Makedon. 2006. Attack detection in time series for recommender systems. In *KDD*. 809–814.

[154] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.* 52, 1 (2019), 1–38.

[155] Shijie Zhang, Hongzhi Yin, Tong Chen, Zi Huang, Quoc Viet Hung Nguyen, and Lizhen Cui. 2022. PipAttack: Poisoning federated recommender systems for manipulating item promotion. In *WSDM*. 1415–1423.

[156] Shijie Zhang, Hongzhi Yin, Tong Chen, Quoc Viet Nguyen Hung, Zi Huang, and Lizhen Cui. 2020. GCN-based user representation learning for unifying robust recommendation and fraudster detection. In *SIGIR*. 689–698.

[157] Yongfeng Zhang, Yunzhi Tan, Min Zhang, Yiqun Liu, Tat-Seng Chua, and Shaoping Ma. 2015. Catch the black sheep: Unified framework for shilling attack detection based on fraudulent action propagation. In *IJCAI*.

[158] Yihe Zhang, Xu Yuan, Jin Li, Jiadong Lou, Li Chen, and Nian-Feng Tzeng. 2021. Reverse attack: Black-box attacks on collaborative recommendation. In *SIGSAC*. 51–68.

[159] Leah Zhang-Kennedy and Sonia Chiasson. 2021. A systematic review of multimedia tools for cybersecurity awareness and education. *Comput. Surv.* 54, 1 (2021), 1–39.

[160] Yaru Zhao, Jianbiao Zhang, and Yihao Cao. 2023. Manipulating vulnerability: Poisoning attacks and countermeasures in federated cloud–edge–client learning for image classification. *Knowl.-based Syst.* 259 (2023), 110072.

[161] Q. Zhou, J. Wu, and L. Duan. 2020. Recommendation attack detection based on deep learning. *Journal of Information Security and Applications*, 52 (2020), 102493.

[162] Wei Zhou, Yun Sing Koh, Junhao Wen, Shafiq Alam, and Gillian Dobbie. 2014. Detection of abnormal profiles on group attacks in recommender systems. In *SIGIR*. 955–958.

[163] Wei Zhou, Junhao Wen, Yun Sing Koh, Shafiq Alam, and Gillian Dobbie. 2014. Attack detection in recommender systems based on target item analysis. In *IJCNN*. 332–339.

[164] Wei Zhou, Junhao Wen, Yun Sing Koh, Qingyu Xiong, Min Gao, Gillian Dobbie, and Shafiq Alam. 2015. Shilling attacks detection in recommender systems based on target item analysis. *PloS One* 10, 7 (2015), e0130968.

[165] Wei Zhou, Junhao Wen, Qiang Qu, Jun Zeng, and Tian Cheng. 2018. Shilling attack detection for recommender systems based on credibility of group users and rating time series. *PloS One* 13, 5 (2018), e0196533.

[166] Wei Zhou, Junhao Wen, Qingyu Xiong, Min Gao, and Jun Zeng. 2016. SVM-TIA a shilling attack detection method based on SVM and target item analysis in recommender systems. *Neurocomputing* 210 (2016), 197–205.

Received 17 October 2022; revised 8 August 2023; accepted 18 June 2024