

Article

A Robust Recommender System Against Adversarial and Shilling Attacks Using Diffusion Networks and Self-Adaptive Learning

Ali Alhwayzee ¹ , Saeed Araban ^{1,*} and Davood Zabihzadeh ² 

¹ Computer Engineering Department, Engineering Faculty, Ferdowsi University of Mashhad (FUM), Mashhad 9177948974, Iran; ali.alhwayzee@mail.um.ac.ir

² Computer Engineering Department, Hakim Sabzevari University, Sabzevar 9617976487, Iran; d.zabihzadeh@hsu.ac.ir

* Correspondence: araban@um.ac.ir

Abstract: Shilling and adversarial attacks are two main types of attacks against recommender systems (RSs). In modern RSs, existing defense methods are hindered by the following two challenges: (1) the diversity of RSs' information sources beyond the interaction matrix, such as user comments, textual data, and visual information; and (2) most defense methods are robust only against specific types of adversarial attacks. Ensuring the robustness of RSs against new adversarial attacks across different data sources remains an open problem. To address this problem, we propose a novel method that unifies adversarial attack detection, purification, and fake user detection in RSs by utilizing a guided diffusion adversarial purification network and a self-adaptive training technique. Our approach aims to simultaneously handle both known and unknown adversarial attacks on RSs' inputs and outputs. We conducted extensive experiments on three large-scale datasets to evaluate the effectiveness of the proposed method. The results confirm that our method can effectively eliminate adversarial perturbations on images and textual content within RSs, surpassing state-of-the-art methods by a significant margin. Moreover, it achieved the best results in three out of five evaluated shilling attack types. Finally, for attacks with realistic magnitudes, it can maintain baseline performance levels even when multiple attacks are applied simultaneously.

Keywords: adversarial attacks; shilling attacks; robust recommender systems; adversarial purification; diffusion networks; self-adaptive training



Academic Editor: Marek T. Malinowski

Received: 8 December 2024

Revised: 16 January 2025

Accepted: 30 January 2025

Published: 5 February 2025

Citation: Alhwayzee, A.; Araban, S.; Zabihzadeh, D. A Robust Recommender System Against Adversarial and Shilling Attacks Using Diffusion Networks and Self-Adaptive Learning. *Symmetry* **2025**, *17*, 233. <https://doi.org/10.3390/sym17020233>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Recommender systems (RSs) are major components in various domains such as e-commerce, search engines, and financial services. They bridge the gap between users and products by presenting personalized content, ensuring a harmonious and symmetric user experience. RSs not only aid customers in finding relevant items efficiently but also directly influence their decision-making processes by providing tailored suggestions. However, growing concerns about RSs may compromise users' trust due to vulnerabilities such as non-transparency, unfair treatment among different users or groups, extensive use of private user data, and the provision of tampered content guided by attackers or fake users.

A holistic definition of a robust RS encompasses several perspectives of robustness, including robustness with respect to sub-populations, distributional shifts, sparsity, and attacks that corrupt features of items, users, and interactions [1]. This research focuses on the robustness of an RS against attacks on its content data and interactions.

Even if an RS ethically collects and employs user data, privacy concerns remain if external attackers breach the system or expose crucial system details such as log data. On the one hand, inadequate security measures and data anonymization within the RS could allow adversaries to retrieve user data and system information through hacking or inference [2]. On the other hand, attackers could impersonate users or trusted third parties interacting with the recommendation system to manipulate its decision-making process. For instance, a fake user might intentionally inject biased data to deceive the RS, thereby steering the recommendation results to favor specific items or business entities while suppressing others [3].

This research examines two main types of attacks against RSs: (1) shilling attacks, and (2) adversarial attacks on content data. Shilling attacks aim to manipulate the user–item interaction matrix by adding fake interactions to influence the predicted ratings [4]. On the other hand, adversarial attacks target the content data related to users and items in an RS.

The majority of research on adversarial attacks against content data has focused on visual information (images) in RSs. However, modern RSs utilize various information sources such as user profiles, user comments, social connections, and textual item information. Therefore, ensuring the robustness of an RS against adversarial attacks on any of these different data sources is an open problem [5]. Additionally, existing methods are only robust against specific types of adversarial attacks, leading to an endless cycle of attacks and defenses [6]. Recently, PORE [6] proposed a solution for this challenge, but it is limited to *untargeted* adversarial attacks against the interaction matrix, *not content data*. Furthermore, there is still room for improvement in the robustness of RSs against novel (unseen) attack types. Some studies, such as [7], focus on enhancing system robustness against multiple simultaneous attacks; however, these approaches are outside the scope of the RS domain.

To address these challenges, this research presents a novel method called *Unified input/target Attack Purifier and Detector* (UAPD), which unifies *adversarial* and *shilling* attack detection and purification, as well as fake user detection in RSs, by utilizing diffusion networks and employing a self-supervised strategy.

Specifically, to detect and purify adversarial examples, we pass each input through a diffusion network that gradually submerges the adversarial perturbations by adding Gaussian noise and then simultaneously removes both types of noise following a guided denoising process. To enhance the purification and detection capabilities of the diffusion network, we first identify the adversarial directions for a set of clean examples and then train the network to generate a symmetric clean variant of each example, regardless of whether it receives an adversarial or clean input.

Additionally, to enhance the robustness of a baseline RS against possibly *malicious user–item interactions*, UAPD adapts the *self-adaptive training strategy* [8], enabling our model to detect and refine the attacked targets during the training process using the model's own predictions. To detect fake user profiles, we first identify potential target items by comparing the outputs of the self-adaptive training for each item with their initial interactions. Items whose updated interactions are significantly different from their initial values are identified as candidate target items. Then, the profiles of users whose behavior on this target set differs significantly from that of other users are classified as fake profiles.

UAPD offers high generalizability against unseen adversarial and shilling attacks due to the following points:

1. The diffusion-based purification method in UAPD can naturally eliminate any adversarial perturbation during the diffusion phase, regardless of the attack type. The theoretical reason for this feature has been discussed in Section 4.4. Moreover, the results of the ablation study (in Section 5.7) also confirm that the proposed purifica-

tion can eliminate adversarial noise effectively without training the DDPMs for any specific attacks.

2. The modified self-supervised training process is generalizable to unseen shilling attacks because it does not rely on any specific type of shilling attack.

We conducted extensive experiments on three large-scale RS benchmarks to assess the effectiveness of UAPD. The results demonstrate that our approach effectively purifies adversarial perturbations from both image and text inputs in RSs, achieving substantial improvements over current state-of-the-art methods. Moreover, it can identify noisy interactions and detect fake user profiles under various shilling attacks. Notably, it outperformed other methods in three out of five evaluated shilling attack scenarios. Furthermore, under attacks with realistic intensity, our method preserves the baseline RS performance even when multiple attacks are applied concurrently.

The main contributions of this paper can be summarized as follows:

- The proposed method unifies adversarial attack detection and purification, as well as fake user detection in RSs.
- It can handle most adversarial attacks on inputs and noisy interactions simultaneously.
- UAPD can identify and purify both known and unknown adversarial attack types in content-based or hybrid RSs.

Tables 1 and 2 present the main abbreviations and notations used throughout this paper. The remainder of this paper is organized as follows: Section 2 reviews related work on adversarial and shilling attacks in RSs. Section 3 provides background information on diffusion networks and the self-adaptive training method. Section 4 presents the proposed method, including training, detection steps, and implementation details. Section 5 presents and analyzes experimental results, compares the proposed method's performance with other methods, and examines various aspects of UAPD through an ablation study. Finally, Section 6 concludes the paper by summarizing the key findings and discussing potential future research directions.

Table 1. Summary of main abbreviations.

Abbreviation	Description
RS	Recommender system
UAPD	The proposed method, a unified input/target attack purifier and detector for RSs
PORE	Provably Robust Recommender System against data poisoning attacks
VBPR	Visual Bayesian Personalized Ranking [9]
DSSM-Vis	Visual DSSM modality-based RS [10]
DSSM-Text	Textual DSSM modality-based RS [10]
NMF	Non-negative matrix factorization
FGSM	Fast Gradient Sign Method attack
PGD	Projected Gradient Descent attack
C&W	Carlini & Wagner attack
DDPM	Denosing Diffusion Probabilistic Model
U-Net	Architecture used in DDPM for denoising
HM	Large-scale personalized fashion recommendations dataset
MIND	Large-scale dataset for news recommendation tasks
Amazon Men	Subset of the Amazon Reviews dataset focusing on men-related products
AT	Adversarial Training [11]
FAT	Free Adversarial Training [11]
AiD	Adversarial Image Denoiser [12]
AMR	Adversarial Multimedia Recommendation [13]

Table 1. Cont.

Abbreviation	Description
PoisonRec	Poisoning Recommender Systems attack [14]
DL_Attack	Deep learning-based attack [15]
RAPU	Robust Adversarial Poisoning with Uncertainty attack [16]
ARLib	Library for Attacks against Recommendation [17]
APT	Adversarial Poisoning Training [18]
Bagging	Defense method that uses ensemble techniques [19]
Target ratio	Proportion of target items relative to total item count
Malicious rate size	Fraction of total items rated by fake users

Table 2. Summary of main notations.

Notation	Description	Notation	Description
T	Number of steps in the forward and reverse processes	T_c	Number of time steps for purifying
D_i, V_i	Description and image of item i	$\hat{D}_i, \hat{V}_i$	Purified description and image for item i
β_t	Small constant denoting noise added at time step t	α_t	Coefficient, defined as $1 - \beta_t$
q, p_θ	Distributions in the forward and reverse processes	ϵ_θ	Estimated noise by the DDPM
$r_{ui} \in R$	The rating or interest of user u for item i	$\hat{r}_{ui}$	Predicted interest score for user u in item i
$w_i^{(D)}, w_i^{(V)}$	Likelihood of adversarial noise in D_i and V_i	$\hat{w}_{ui}$	Estimating the probability that r_{ui} is clean
$\mu_\theta, \Sigma_\theta$	Predicted mean and covariance for the reverse process at time step t	δ	Adversarial perturbation
λ_{EMA}	Smoothing parameter in EMA	$\mathcal{L}_D$	Prediction's consistency loss term
γ_t	Guidance scale at time step t	$\mathcal{L}_{DDPM}$	DDPM's training loss
$\mathcal{U}$	List of n_U users	$\mathcal{I}$	Set of n_I items
U_i^+	Set of users who positively interacted with item i	I_u^+	Set of items positively interacted with by user u
R	User-item interaction matrix, $R \in [0, 1]^{n_U \times n_I}$	I_{tar}	Set of identified target items
$\mathcal{L}_{RS}$	Loss function of the baseline RS	$\mathcal{L}_f$	Final loss function
τ	Threshold for identifying a fake user	τ_i	Threshold for labeling an item as a target item
ϵ	Adversarial perturbation size	η	Fake user fraction, percentage of fake users in the dataset
$F(i)$	Metric for identifying target items	$F(u)$	Metric for quantifying the fakeness of user u
HR@K	Hit Ratio at top-K ranking, $K \in \{10, 20\}$	NDCG@K	Normalized Discounted Cumulative Gain at top-K ranking
IDCG _u	Ideal DCG for user u	RI(m)	Robustness improvement for metric m

2. Related Work

In this section, we review the recent trends and advancements to improve the robustness of RSs against shilling and machine learning adversarial attacks.

2.1. Shilling Attacks Methods

Shilling attacks on RSs have been extensively studied in the literature. The initial studies focused on creating artificial profiles with specific rating strategies, such as random [20], popular [21], love–hate [22], bandwagon [22], and average [20], among others. The aim of these attacks was to manipulate the user–item interaction matrix by adding fake interactions to influence the recommendation system’s output and achieve malicious goals [4].

There are two main approaches to tackle the consequences of shilling attacks. The first approach involves utilizing a shilling attack detection algorithm to identify and then eliminate the attack profiles from the interaction matrix. The second approach is to develop recommendation methods that are resistant to shilling attacks. Table 3 provides a summary of various attack and defense methods related to shilling attacks.

Table 3. Summary of attack and defense methods against shilling attacks.

Methods	Description
Shilling Attack Evaluation Methods [4,20–22]	Evaluated four types of shilling attacks on RSs: random, average, bandwagon, and segment attacks.
Supervised Shilling Attack Detection Methods [3,6,23–25]	Handled specific rating strategies or machine learning data poisoning attacks on the interaction matrix.
Unsupervised Shilling Attack Detection Methods [26,27]	Detected attacks by clustering or propagation methods.
Robust RSs vs. Shilling Attacks [28,29]	Improved robustness of RSs using regularization terms, kernel methods, or generative models.

Numerous methods have been developed to detect shilling attacks [3,4,23–27,30–32]. These methods can be classified into three groups: supervised classification, unsupervised clustering, and robust RSs.

2.1.1. Supervised Shilling Attack Detection Methods

Supervised classification methods typically involve feature engineering, followed by the development of a detection algorithm using the extracted features. For example, Yang et al. [23] extracted three features from user profiles based on filler size. The filler size of a user is defined as the number of items rated by the user among all the items in the RS. They then examined these features using statistical tests and applied a modified version of the AdaBoost algorithm, called *Re-scale AdaBoost*, to identify attack profiles. Moreover, other studies extracted various features from rating scores to train a binary classifier to distinguish between normal users and fake users. Examples of these features include *rating deviation from mean agreement*, *weighted degree of agreement*, *weighted deviation from mean agreement*, *mean-variance*, and *filler mean target difference* [3].

Fang et al. [24] addressed data poisoning optimization attacks for top-N recommendations, where attackers aim to promote a particular item to appear in the top lists of as many normal users as possible. Hu et al. [25] explored two different attacks: (1) injecting fake ratings and (2) generating fake relationships with normal users. They assumed that the

RS recommends a list of N items with the highest rating scores to users, and that attackers aim to maximize the hit probability of the target item in the recommendation list. Since directly formulating the hit probability of an item is challenging, they, similar to previous work [3], approximated it using the *Wilcoxon–Mann–Whitney loss* [32]. Here, the loss was used to statistically compare the ranks of the target item and non-target items in the RS. A significant difference between the ranks implies a potential change in the hit probability of the target item, which can be used to estimate the success of the attacks.

Jia et al. [6] proposed PORE as the first provably robust RS against data poisoning attacks. PORE can be applied to existing RSs to make them robust against untargeted data poisoning attacks, which are designed to reduce the overall performance of an RS. To achieve this, PORE guarantees that a certain number of top recommendations (e.g., r out of N items) will remain accurate for users, even in the presence of such attacks. Here, the value of r depends on the number of fake users involved in the attack.

A Time and Trust Recommender System (T&TRS) [33] incorporates a community detection algorithm that leverages explicit and implicit trust relationships among users, as well as the temporal dynamics of ratings, to identify and mitigate the influence of shilling attacks. However, the reliance on explicit trust data may limit the applicability of T&TRS in platforms lacking such information, and the computational complexity associated with community detection in large-scale systems requires further consideration.

Large Language Model for Robust Sequential Recommendation (LoRec) [34] enhances the resilience of sequential recommender systems against poisoning attacks by incorporating an LLM-enhanced Calibrator (LCT). An LCT refines the training process through user reweighting, thereby mitigating the effects of fraudulent user profiles. While this approach appears effective against shilling attacks, its reliance on LLMs raises concerns about computational overhead and resource requirements. Additionally, the integration of an LLM makes it harder to interpret how LoRec assigns weights to different users because LLMs function as black box models.

2.1.2. Unsupervised Shilling Attack Detection Methods

Unsupervised methods typically involve clustering user profiles and then eliminating suspicious clusters. For example, Bhaumik et al. [26] clustered user profiles using the k -means algorithm and labeled small clusters as attack groups. Similarly, Li et al. [27] estimated the likelihood of each user being a member of an attack group using a propagation approach. This approach calculates the likelihood of a user based on the user's connections to a set of trustworthy users in a baseline RS.

2.1.3. Robust RSs Against Shilling Attacks

Alternatively, some methods have been proposed to improve the robustness of RSs against shilling attacks. For example, Zhang et al. [28] developed a robust collaborative filtering method by adding an L1-norm regularization term to non-negative matrix factorization (NMF). Similarly, Li and Luo [29] presented a robust matrix factorization method using kernel techniques.

To mitigate the effects of noisy user–item interactions, the *Diffusion Recommender Model* (DiffRec) [35] uses diffusion networks to gradually corrupt user interaction histories with Gaussian noise and iteratively reconstruct the original interactions. Similarly, ref. [36] enhances the robustness of user and item embeddings in recommender systems against noisy implicit feedback. It employs a multi-step denoising approach based on diffusion models that iteratively injects and removes Gaussian noise from embeddings. While these methods help the baseline RS mitigate the effects of noisy interactions, the RS remains vulnerable to adversarial shilling attacks.

2.2. Adversarial Attacks on Content Data

This group of attacks targets content data related to users and items in RSs and is primarily used against content-based or hybrid RSs. An example is the DeepWordBug attack presented in ref. [37], which is a black box attack that efficiently generates adversarial text sequences by minimally modifying original texts to deceive a deep *Recurrent Neural Network* (RNN)-based classifier.

The majority of work against this type of attack has focused on visual information (images) in RSs. For instance, by adding an untargeted adversarial perturbation to a single product photo, ref. [13] showed that the *Visual Bayesian Personalized Ranking* (VBPR) [9] assigned this product a significantly lower ranking than before. Similarly, the effect of targeted adversarial attacks such as the *Fast Gradient Sign Method* (FGSM) and *Projected Gradient Descent* (PGD) on a *Conventional Neural Network* (CNN)-based visual RS was studied by ref. [38]. They found that the system incorrectly assigned low-popularity products to the category of high-demand items. Moreover, by making a small, imperceptible change to an image, the product's chance of being suggested increased by more than 300%.

Tang et al. [13] employed Adversarial Training strategies to handle these attacks. However, this approach was only effective against attacks observed during the training phase and was limited to item images, neglecting other information sources in RSs.

A few studies have focused on content-based adversarial attacks on data types other than images, such as text, audio, and video. For example, ref. [39] proposed a method for generating adversarial audio samples that target a specific transcription while keeping the original sound perceptually intact.

3. Preliminaries

3.1. Diffusion Network

Let p be the distribution of clean input data and let $p_z(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mathbf{0}, \mathbf{I})$ be the latent distribution. For example, $p(\mathbf{x})$ denotes the distribution of item images without adversarial perturbations, and $p_z(\mathbf{x}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ represents the corresponding latent distribution. A *Denoising Diffusion Probabilistic Model* (DDPM) is composed of two Markov chains: the *forward* (diffusion) process and the *reverse* (denoising) process. The diffusion process gradually adds Gaussian noise to the clean data until they become approximately pure random Gaussian noise. In contrast, the reverse process progressively denoises the random Gaussian noise to restore a symmetric sample from the original clean data distribution. Both processes have the same number of steps, denoted by T .

Specifically, let $\mathbf{x} = \mathbf{x}^0 \sim p(\mathbf{x})$. The forward process is defined as follows:

$$q(\mathbf{x}^t|\mathbf{x}^{t-1}) = \mathcal{N}(\mathbf{x}^t|\sqrt{\alpha_t}\mathbf{x}^{t-1}, \beta_t\mathbf{I}), \quad (1)$$

where $\beta_t \geq 0$ is a small constant denoting the added noise at time step t , and $\alpha_t = 1 - \beta_t$. The cumulative noise addition to the original data over t steps can be described as follows:

$$q(\mathbf{x}^t|\mathbf{x}^0) = \mathcal{N}(\mathbf{x}^t|\sqrt{\bar{\alpha}_t}\mathbf{x}^0, (1 - \bar{\alpha}_t)\mathbf{I}) \quad (2)$$

where $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$. Equivalently, we have the following:

$$\mathbf{x}^t = \sqrt{\bar{\alpha}_t}\mathbf{x}^0 + \sqrt{(1 - \bar{\alpha}_t)}\boldsymbol{\epsilon}, \text{ where } \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \quad (3)$$

When T is large, $\bar{\alpha}_T$ approaches zero. Consequently, according to Equation (2), $q(\mathbf{x}^T|\mathbf{x}^0)$ converges to $p_z(\mathbf{x}^T) = \mathcal{N}(\mathbf{0}, \mathbf{I})$.

The reverse process reconstructs the original input $\mathbf{x}^0$ from the noisy data $\mathbf{x}^T$ through a series of denoising steps. To this end, the process uses a trained neural network to predict the mean $\boldsymbol{\mu}_\theta(\mathbf{x}^{(t+1)}, t)$ and covariance $\boldsymbol{\Sigma}_\theta(\mathbf{x}^{(t+1)}, t)$ for the distribution of $\mathbf{x}^t$ given $\mathbf{x}^{(t+1)}$. The noise is removed step-by-step until $\mathbf{x}^0$ is recovered. The distribution is approximated as follows:

$$p_\theta(\mathbf{x}^t | \mathbf{x}^{(t+1)}) = \mathcal{N}(\mathbf{x}^t | \boldsymbol{\mu}_\theta(\mathbf{x}^{(t+1)}, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}^{(t+1)}, t)). \quad (4)$$

To generate a symmetric new example, a DDPM begins with $\mathbf{x}^T \sim p_z(\mathbf{x})$. It then samples $\mathbf{x}^t \sim p_\theta(\mathbf{x}^t | \mathbf{x}^{(t+1)})$ ($t = T - 1, \dots, 1, 0$) and ultimately returns $\mathbf{x}^0$ as the output.

3.2. Self-Adaptive Training

Self-adaptive training introduces a unified training algorithm that dynamically calibrates and enhances the training process of deep neural networks using the model's predictions. It is based on the observation that deep models can identify useful patterns from input data by using their own predictions in both supervised and self-supervised training approaches.

Formally, let $\mathcal{X} = \{(\mathbf{x}_i, \mathbf{y}_i) : i \in 1, 2, \dots, N\}$ be the input dataset with potentially noisy labels. This method tracks the model's predictions on the dataset at each training step:

$$\mathbf{p}_i^{(m)} = p_m(\mathbf{y} | \mathbf{x}_i) \in [0, 1]^K, \quad i \in 1, 2, \dots, N. \quad (5)$$

Here, $\mathbf{p}_i^{(m)}$ denotes the probability distribution over the K classes predicted by the model for the input $\mathbf{x}_i$. The targets are then updated using an *Exponential Moving Average* (EMA) scheme:

$$\tilde{\mathbf{y}}_i \leftarrow \lambda_{EMA} \mathbf{p}_i^{(m)} + (1 - \lambda_{EMA}) \tilde{\mathbf{y}}_i. \quad (6)$$

Let $\tilde{\mathbf{y}}_i$ represent the modified label, which is initially set equal to $\mathbf{y}_i$. This approach gradually shifts $\tilde{\mathbf{y}}_i$ towards the model's current prediction $\mathbf{p}_i^{(m)}$, where λ_{EMA} is a smoothing hyperparameter that determines the weight of the new prediction versus the current label. It allows the training process to completely change a label if the model's predictions consistently differ from the original noisy label. This approach was initially proposed for classification tasks; however, we extended it to recommender models to modify the interest of a user toward an item, as described in Section 4.5.

4. The Proposed Method

4.1. Formalizing the Problem

Let $\mathcal{U}$ be the list of n_U users, $\mathcal{I}$ denote the item set including n_I items, and the matrix $R \in [0, 1]^{n_U \times n_I}$ be the user-item interaction matrix, where a nonzero $r_{ui} \in R$ denotes the rating or interest of user u for item i scaled to the range $(0, 1]$. $I_u^+ \subseteq \mathcal{I}$ denotes the set of items toward which user u has positive interactions (e.g., 4 or 5, or items purchased by u).

For simplicity, we assume that each user is represented by a unique identifier, and each item, in addition to a unique identifier, is associated with multi-modal information: D_i and V_i , which represent the item's description and image, respectively. However, our method can be extended to capture other sources of information, such as item reviews and user comments.

The target of an RS is typically a score value r_{ui} , indicating the interest of a given user u in item i . We assume that the target is scaled in the range $[0, 1]$. A rating or ranking-based RS predicts the rating or interest r_{ui} of a user u in an item i :

$$\hat{r}_{ui} = p_m(X_{ui}) \in [0, 1], \text{ where } X_{ui} = \{u, i, R, D_i, V_i\}. \quad (7)$$

where p_m denotes the model's prediction for the given inputs. Moreover, we can encode the target similar to one-hot encoding as follows:

$$\hat{r}_{ui} = \mathbf{p}_m(X_{ui}) = [1 - \hat{r}_{ui}, \hat{r}_{ui}] \in [0, 1]^2. \quad (8)$$

We assume that the elements of matrix R (targets) or any of the recommender system's inputs, like an item's image or description, can be changed or perturbed by attackers for unknown adversarial objectives.

4.2. Overview of the Proposed Method

The proposed method enhances the robustness of a baseline RS, which could be any existing multi-modal RS, such as those introduced in refs. [9,10,40].

Figure 1 illustrates the architecture and the main training stages of the proposed method. Prior to training the baseline RS, our approach performs a novel pretraining step that fits two guided DDPMs, thereby enhancing their ability to remove adversarial perturbations from their respective inputs.

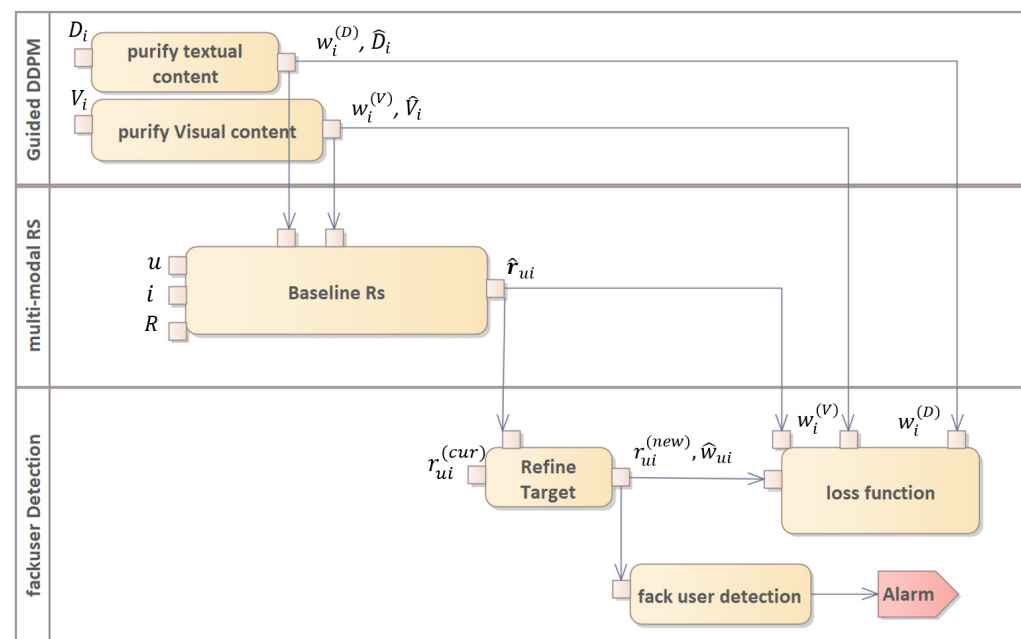


Figure 1. Architecture and main training steps of the proposed method.

Given $(\langle u, i, R, D_i, V_i \rangle, r_{ui})$, UAPD first purifies the textual content D_i and the visual content V_i using the pretrained DDPMs, producing purified contents $\hat{D}_i$ and $\hat{V}_i$ along with corresponding weights, $w_i^{(D)}$ and $w_i^{(V)}$, that quantify the likelihood of adversarial attacks on the item's textual and visual information. Subsequently, UAPD provides the purified input $\langle u, i, R, \hat{D}_i, \hat{V}_i \rangle$ and the weights $w_i^{(D)}$ and $w_i^{(V)}$ to the baseline RS, obtaining $\hat{r}_{ui}$, which estimates the interest of user u in item i .

The target r_{ui} is then modified by the self-adaptive training mechanism, which also assigns a weight $\hat{w}_{ui}$ that estimates the probability that r_{ui} is clean. Finally, the loss function of the baseline RS is computed using the modified target r_{ui} , the weights $w_i^{(D)}$, $w_i^{(V)}$, and $\hat{w}_{ui}$, and the model's estimation $\hat{r}_{ui}$. The resulting gradient is then propagated to update the baseline RS's parameters.

Additionally, we introduce a fake user detection module that identifies fake users by comparing modified and initial target values for each user. In the following sections, we describe each stage of UAPD in detail.

4.3. Training DDPMs

The proposed method trains a DDPM for each modality of input data (item description and item image). Let x denote an input from an arbitrary modality (i.e., $x = D_i$ or $x = V_i$). To enhance the purification power of the DDPM, we pretrain it on a clean dataset so that it learns to generate a sample similar to the clean input x by receiving either x or an adversarially perturbed variant of x , denoted as $x^{(a)}$. This process is illustrated in Figure 2.

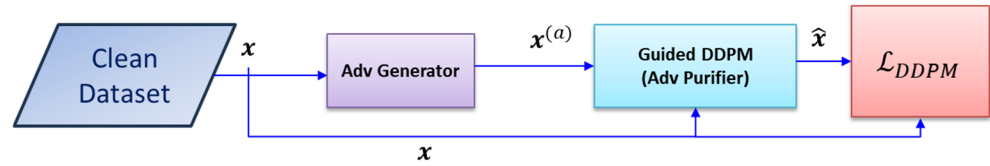


Figure 2. Training steps of a DDPM on a clean dataset.

More precisely, we obtain an adversarial direction δ by solving the following optimization problem:

$$\delta = \arg \max_{\delta: \|\delta\|_2 \leq \epsilon_{adv}} \mathcal{L}_D(p_m(x + \delta), p_m(x)), \quad (9)$$

where $\mathcal{L}_D$ measures the distance between the model's predictions on $x + \delta$ and x . In practice, $\mathcal{L}_D$ can be modeled by a regression loss such as MSE, Huber loss, or the Binary Cross Entropy classification loss. The adversarial perturbation δ can be approximated by ref. [41]:

$$\delta \approx \frac{\epsilon_{adv} g}{\|g\|}, \text{ where } g = \nabla_x \mathcal{L}_D(p_m(x + \delta), p_m(x)) \quad (10)$$

This approximation follows the linearization of the loss function $\mathcal{L}_D$ around x . It assumes that δ is small, allowing higher-order terms in the Taylor expansion to be neglected [41].

Subsequently, we obtain $x^{(a)} = x + \delta$ and pass both x and $x^{(a)}$ to the network to optimize its parameters using the proposed hybrid loss, which is composed of a reconstruction loss term such as *Mean Square Error* (MSE) and a prediction's consistency term $\mathcal{L}_D$:

$$\mathcal{L}_{DDPM}(v, x, \hat{v}) = \|x - \hat{v}\|_2^2 + \mathcal{L}_D(p_m(x), p_m(\hat{v})) \text{ where } v \in \{x, x^{(a)}\} \quad (11)$$

To train the DDPM using $\mathcal{L}_{DDPM}$, we first generate a random timestep $t \in (1, 2, \dots, T_c)$ and sample a Gaussian noise $\epsilon \sim \mathcal{N}(\mathbf{0}, I)$. Then, we diffuse v for t steps and obtain v^t :

$$v^t = \sqrt{\bar{\alpha}_t} v + \sqrt{(1 - \bar{\alpha}_t)} \epsilon$$

Then, the network estimates the noise ϵ_θ and reconstructs $\hat{v}$ using the estimated noise ϵ_θ as follows:

$$\hat{v} = \hat{v}^0 = \frac{1}{\sqrt{\bar{\alpha}_t}} v^t - \frac{\sqrt{(1 - \bar{\alpha}_t)}}{\sqrt{\bar{\alpha}_t}} \epsilon_\theta \quad (12)$$

Afterward, the loss is computed between $\hat{v}$ and x , which encourages the DDPM to eliminate both the added Gaussian noise and the adversarial perturbation simultaneously.

4.4. Adversarial Purification and Detection

A DDPM can naturally eliminate any adversarial perturbation in the diffusion phase by gradually adding Gaussian noises to the input data. Then, it can achieve a symmetric clean input from the output of the diffusion phase through the reverse process. Specifically, given an adversarial example $\mathbf{x}^{(a)} = \mathbf{x} + \delta$ with adversarial perturbation δ , if we diffuse $\mathbf{x}^{(a)}$ for T_c steps, we observe the following:

$$\mathbf{x}^{T_c} = \sqrt{\bar{\alpha}_{T_c}}(\mathbf{x} + \delta) + \sqrt{(1 - \bar{\alpha}_{T_c})} \epsilon. \quad (13)$$

As T_c increases, the coefficient $\sqrt{\bar{\alpha}_{T_c}}$ decreases and $\sqrt{1 - \bar{\alpha}_{T_c}}$ increases. Moreover, $\|\delta\| \ll \|\mathbf{x}\|$, since δ should be perceptually indistinguishable. Therefore, we can select T_c such that the added Gaussian noise $\sqrt{1 - \bar{\alpha}_{T_c}}\epsilon$ becomes large enough to submerge the weakened adversarial perturbation $\sqrt{\bar{\alpha}_{T_c}}\delta$, while the main content of the input data is preserved simultaneously.

As seen, a trade-off exists between the purification effect and consistency with the original clean input data. If we use a large value for T_c , the purified data will deviate from the original data. Moreover, a small value for T_c causes the adversarial perturbation to remain in the purified data. This implies the need for an approach that allows us to use a large number of diffusion steps (T_c) to achieve effective purification while still generating purified data close to the original data.

To achieve this, we use the input data to guide the reverse process to generate a sample similar to the original data. In particular, we condition each reverse denoising step on the input data by changing the distribution $p_\theta(\mathbf{x}^t | \mathbf{x}^{(t+1)})$ to $p_\theta(\mathbf{x}^t | \mathbf{x}^{(t+1)}, \mathbf{x}^{0t})$, where $\mathbf{x}^{0t}$ is obtained by applying t diffusion steps to the original input data (i.e., $\mathbf{x}^0$):

$$\mathbf{x}^{0t} = \sqrt{\bar{\alpha}_t}\mathbf{x}^0 + \sqrt{(1 - \bar{\alpha}_t)}\epsilon \quad (14)$$

For brevity, let $\boldsymbol{\mu} = \boldsymbol{\mu}_\theta(\mathbf{x}^{(t+1)}, t)$ and $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}_\theta(\mathbf{x}^{(t+1)}, t)$. Similarly to ref. [42], $p_\theta(\mathbf{x}^t | \mathbf{x}^{(t+1)}, \mathbf{x}^{0t})$ can be expressed as follows:

$$p_\theta(\mathbf{x}^t | \mathbf{x}^{(t+1)}, \mathbf{x}^{0t}) \approx \mathcal{N}(\mathbf{x}^t | \hat{\boldsymbol{\mu}}_\theta(\mathbf{x}^{(t+1)}, \mathbf{x}^{0t}, t), \boldsymbol{\Sigma}) \quad (15)$$

where

$$\hat{\boldsymbol{\mu}}_\theta(\mathbf{x}^{(t+1)}, \mathbf{x}^{0t}, t) = \boldsymbol{\mu} + \boldsymbol{\Sigma} \nabla_{\mathbf{x}^t} \ln p(\mathbf{x}^{0t} | \mathbf{x}^t) \big|_{\mathbf{x}^t = \boldsymbol{\mu}}. \quad (16)$$

The proof is presented in Appendix A. Similarly to ref. [43], we can approximate $p(\mathbf{x}^{0t} | \mathbf{x}^t)$:

$$p(\mathbf{x}^{0t} | \mathbf{x}^t) = \frac{1}{Z} \exp(-\gamma_t d(\mathbf{x}^t, \mathbf{x}^{0t})). \quad (17)$$

Here, $d(\mathbf{x}^t, \mathbf{x}^{0t})$ is a distance measure like $\|\mathbf{x}^t - \mathbf{x}^{0t}\|_2^2$, Z is the normalizing constant, and γ_t is the guidance scale at time step t . Consequently, we can approximate the conditional distribution $p_\theta(\mathbf{x}^t | \mathbf{x}^{(t+1)}, \mathbf{x}^{0t})$ with a normal distribution like the standard DDPM. The key difference is that the mean is shifted by $\gamma_t \boldsymbol{\Sigma} \nabla_{\mathbf{x}^t} \ln p(\mathbf{x}^{0t} | \mathbf{x}^t) \big|_{\mathbf{x}^t = \boldsymbol{\mu}}$.

After purifying the input data using the guided DDPM, we can detect the existence of adversarial attacks on the input. Our detection mechanism relies on the observation that the predictions for adversarial data differ significantly from those for the corresponding clean data.

Let $\hat{x}$ be the purified version of the input x . We compute the model's predictions, $\mathbf{o}$ and $\hat{\mathbf{o}}$, for x and $\hat{x}$, respectively. The prediction consistency loss $\mathcal{L}_D$ between $\mathbf{o}$ and $\hat{\mathbf{o}}$ (i.e., $\mathcal{L}_D(\mathbf{o}, \hat{\mathbf{o}})$) measures the difference between the predictions. Thus, we assign a weight to the input x from a text or visual modality (i.e., D or V) as follows:

$$w^{(M)} = \exp(-\gamma \mathcal{L}_D(\mathbf{o}, \hat{\mathbf{o}})) \quad (18)$$

where $M \in \{D, V\}$ denotes the modality of the input x , and the hyperparameter γ controls the exponential decay rate.

4.5. Refining Targets

In addition to the inputs of an RS, its targets, such as the rating matrix, can be manipulated by malicious users to achieve their own goals. To effectively train the model with potentially noisy targets and restore the true value of these noisy targets, we extend the *self-adaptive training strategy* [8] to recommender systems, allowing the model to progressively refine noisy targets by using its own predictions as guidance during training.

Let r_{ui} be the target of the RS, $\hat{r}_{ui}$ denote the corresponding prediction (output) of the model, and $\tilde{r}_{ui}$ represent the modified target initially set equal to r_{ui} . At each training iteration, we update the target by using the *Exponential Moving Average* (EMA) mechanism as follows:

$$\tilde{r}_{ui} = \lambda_{EMA} \tilde{r}_{ui} + (1 - \lambda_{EMA}) \hat{r}_{ui} \quad (19)$$

where the momentum coefficient λ_{EMA} controls the weight of the model's predictions.

The EMA scheme mitigates the instability of the predictions and smoothly changes the targets if necessary. Moreover, we assign a weight to each target as follows:

$$\hat{w}_{ui} = \max\{\tilde{r}_{ui}, 1 - \tilde{r}_{ui}\} \quad (20)$$

The value of $\hat{w}_{ui}$ reveals the confidence of the corresponding target. Intuitively, during the initial E_S epochs, all examples are treated with equal importance. As the target value $\tilde{r}_{ui}$ is updated, our method reduces its attention to potentially erroneous data and attends more to the potentially clean data. This approach also permits incorrect targets to regain attention if they are confidently refined.

4.6. Loss Function

The loss function plays a key role in the success of our method. It should encourage the model to accurately predict the refined targets. Many SOTA recommender systems such as [9,10,40] sample triplets in the form $\langle u, i, j \rangle$ from the training set, where $\langle u, i \rangle \in I_u^+$ denotes a positive pair and $\langle u, j \rangle$ represents a negative pair typically obtained by random sampling from the set of unobserved items ($\mathcal{I} / I_u^+$). In this research, we sample a negative item j from either unobserved items or items explicitly rated lower by the user.

The loss function of SOTA RSs [9,40] is commonly either the pairwise loss ($\mathcal{L}_{BPR}$) originally introduced in *Bayesian Personal Ranking* (BPR) [44] or *Binary Cross Entropy* (BCE) loss ($\mathcal{L}_{BCE}$) [10]:

$$\mathcal{L}_{BPR}(u, i, j) = -\ln(\sigma(\hat{r}_{ui} - \hat{r}_{uj})) \quad (21)$$

$$\mathcal{L}_{BCE}(u, i, j) = -\ln(\hat{r}_{ui}) - \ln(1 - \hat{r}_{uj}) \quad (22)$$

Let $\mathcal{L}_{RS}(\cdot)$ be the loss function of a baseline recommender system, which is either $\mathcal{L}_{BPR}(\cdot)$ or $\mathcal{L}_{BCE}(\cdot)$. In the proposed method, we extend this loss by incorporating the weights obtained from the purification module $(w_i^{(D)}, w_i^{(V)})$ and the self-adaptive training mechanism $(\hat{w}_{ui})$ for the positive pair. Furthermore, we pass the purified contents $\hat{D}_i$ and $\hat{V}_i$ to the baseline RS. The final loss function is defined as follows:

$$\mathcal{L}_f(u, i, j) = \hat{w}_{ui} w_i^{(V)} w_i^{(D)} \mathcal{L}_{RS}(u, i, j) \quad (23)$$

We experimentally found that combining the weight values using the product rule is more effective than weighted averaging. The product rule assigns a high weight to an interaction when both the inputs and target values of the interaction are detected as clean. Additionally, it avoids introducing extra hyperparameters to control the relative importance of the weights, unlike weighted averaging.

4.7. Detecting Fake Users

The amount of change in a target variable during the target refinement process can serve as a key metric for distinguishing erroneous targets from clean ones. We extend this concept to detect fake users, as we expect that the amount of change in the target values of these users will be much larger than that of normal users.

To this end, we first identify potential target items by comparing the outputs of the *self-adaptive training strategy* for each item with their initial interactions. Specifically, let $R_i = \{r_{ui}\}_{u \in U_i^+}$ and $\tilde{R}_i = \{\tilde{r}_{ui}\}_{u \in U_i^+}$ be the set of original and refined target values of item i , respectively. We identify potential target items by computing the prediction consistency loss $\mathcal{L}_D$ between corresponding targets in two sets, that is,

$$F(i) = \frac{1}{|U_i^+|} \sum_{u \in U_i^+} \mathcal{L}_D(r_{ui}, \tilde{r}_{ui}). \quad (24)$$

We then label any item with $F(i)$ greater than threshold τ_i as a target item. Subsequently, user profiles whose behavior on this target set deviates significantly from others are classified as fake profiles. Specifically, let I_{tar} be the set of identified target items. Moreover, let $R_u = \{r_{ui}\}_{i \in I_u^+ \cap I_{tar}}$ and $\tilde{R}_u = \{\tilde{r}_{ui}\}_{i \in I_u^+ \cap I_{tar}}$ represent the set of original and refined target values of user u over the potential target items, respectively. We quantify the fakeness of user u by computing the prediction consistency loss $\mathcal{L}_D$ between the corresponding targets in these sets, that is,

$$F(u) = \frac{1}{|I_u^+ \cap I_{tar}|} \sum_{i \in I_u^+ \cap I_{tar}} \mathcal{L}_D(r_{ui}, \tilde{r}_{ui}). \quad (25)$$

We identify malicious users using an appropriate threshold value. Specifically, any user with $F(u)$ exceeding threshold τ is labeled as a fake user, and their interactions are excluded from subsequent training epochs. The threshold τ is set as the radius of the minimum hypersphere that encloses at least 95% of the normal user profiles from the validation set. This approach reduces false positives while allowing for the robust detection of fake user profiles.

Algorithm 1 summarizes the main steps of UAPD.

Algorithm 1. UAPD's Training Algorithm**Input:** Train-Set, $DDPM_{Vis}$, $DDPM_{Txt}$: Trained DDPMs, RS_B : Baseline RS**Output:** RS_B : Trained Baseline RS**begin**Initialize modified targets $\{\tilde{r}_{ui}\}$ equal to original r_{ui} **for** $epoch = 1, 2, \dots, \text{MAX_EPOCH}$ **do** **for** each minibatch B in Train-Set **do** $(u, i, r_{ui}) = B$ {Extract user ids, positive item ids, and modified targets from B} $w_i^{(D)} \hat{D}_i = DDPM_{Txt}. \text{purify}(D_i)$ $w_i^{(V)}, \hat{V}_i = DDPM_{Vis}. \text{purify}(V_i)$ $\hat{r}_{ui} = RS_B(u, i)$ $\tilde{r}_{ui} = \lambda_{EMA} \tilde{r}_{ui} + (1 - \lambda_{EMA}) \hat{r}_{ui}$ $\hat{w}_{ui} = \max\{\tilde{r}_{ui}, 1 - \tilde{r}_{ui}\}$ $j = \text{generate negative pairs according to } (u, i)$ $\hat{r}_{uj} = RS_B(u, j)$ $L = \hat{w}_{ui} w_i^{(V)} w_i^{(D)} \mathcal{L}_{RS}(u, i, j)$ Backpropagate L to update RS_B 's parameters **end for****end for****return** RS_B **4.8. Computational and Space Overhead**

To reduce the computational overhead of UADP, we purify the image and description of any item only once and store the purified contents. This increases the space required for storing items by a factor of two. The purification time of images due to the iterative denoising process in DDPMs is relatively time-consuming. However, it demands much less computational time compared to the image generation process by DDPMs. While the number of denoising steps in the DDPM [42] (https://openaipublic.blob.core.windows.net/diffusion/march-2021/imagenet64_cond_270M_250K.pt (accessed on 2 January 2025)) used in our work is 1000 ($T = 1000$), for purifying images, we set the number of denoising steps (T_c) to 36. In our experiments, the required time for purifying a minibatch of images using a single T4 GPU by the pretrained DDPM [42] was about 2.371 s, which could be significantly decreased by increasing the number of GPUs and using more powerful GPUs.

The self-supervised target refinement process needs a lower computational overhead because it only includes the weighting and updating of targets using Equations (19) and (20), adding a small constant to the processing time of a minibatch in the training phase. However, this process requires saving the refined targets, which imposes a space complexity of order $O(n_u)$ because each user interacts positively with only a fixed number of items.

5. Experimental Results

This section describes the experiments conducted to evaluate the effectiveness of the proposed defense method against various adversarial attacks on both the content and interaction matrix of selected baseline RSs, including *Visual Bayesian Personalized Ranking* (VBPR) [9], the *visual DSSM modality-based recommender system* (DSSM-Vis) [10], and the *textual DSSM modality-based recommender system* (DSSM-Text) [10].

We also compare our work with peer defense methods on several real-world datasets: the clothing purchase dataset from the H&M platform (HM) (<https://www.kaggle.com/competitions/h-and-m-personalized-fashion-recommendations/overview> (accessed on 2 January 2025)), the news clicks dataset from the Microsoft news recommendation platform

(MIND) [45], and the Amazon Men dataset, which is a subset of the larger Amazon Reviews dataset, focusing on men-related products [9].

5.1. Datasets

HM is a large-scale personalized fashion recommendations dataset. The dataset provides color, category, and image data for each product, as well as the age, loyalty status, and purchase history for each user.

MIND is a large-scale dataset from Microsoft used for the news recommendation task. It contains news articles and user interactions, such as clicks and browsing behavior. The dataset provides title, abstract, category, subcategory, and text content for each news article.

The Amazon Men dataset, a subset of the larger Amazon Reviews dataset, includes items specifically targeted toward men, such as clothing, accessories, and grooming items. It contains 34,212 users, 100,654 items, and 260,352 interactions. This dataset is commonly used to evaluate visual recommender system models.

Following ref. [10], we randomly selected 500 K and 630 K users from the HM and MIND datasets, along with their referenced items. Users with fewer than 10 interactions were removed from the datasets to filter out extremely sparse user profiles. To construct the interaction matrix, we considered the last 13 and 23 interactions for each selected user in HM and MIND, respectively, to emphasize the most relevant user behavior. We chose to limit the interactions to the last 13 in HM because the task of encoding images demands significantly higher GPU resources. Following ref. [9], users with fewer than ten interactions were also excluded from the Amazon Men dataset, and only the most recent 20 interactions for each user were considered.

For each user, the last and second-to-last referenced items were selected for test and validation purposes, and the remaining referenced items were used for the training set. In HM and Amazon Men, we represented items by images, while in MIND, we represented items by their titles. We considered only the user ID to represent users.

5.2. Evaluation Metrics

We evaluated competing methods using two standard top-K ranking metrics: *Hit Ratio* (HR)@K and *Normalized Discounted Cumulative Gain* (NDCG)@K, where $K \in \{10, 20\}$.

HR@K measures the proportion of times the true relevant item appears within the top-K recommended items. Specifically, let i_u be the ground truth item for user u . HR@K can be expressed as follows:

$$\text{HR@K} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbb{I}(i_u \in \text{top-K recommended items for } u). \quad (26)$$

Let rel_k indicate the binary relevance score of position $k \in [K]$, which is 1 if i_u appears at position k and 0 otherwise. NDCG@K evaluates the ranking quality of the recommended items by examining the positions of relevant items within the top-K list. It assigns higher scores to hits that appear earlier in the ranking.

$$\text{NDCG@K} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{\text{IDCG}_u} \sum_{k \in [K]} \frac{rel_k}{\log_2(k+1)}, \quad (27)$$

where IDCG_u represents the highest possible DCG that can be achieved for the top K ranked items for user u .

In some experiments, we measured the performance of RSs using *robustness improvement* (RI) [18]. For a metric m , such as HR@K and NDCG@K, let $m^{(\text{org})}$, $m^{(\text{def})}$, and $m^{(\text{att})}$

denote the value of m in standard (no attack), defense, and attack without defense settings, respectively. Then, $RI(m)$ is defined as follows:

$$RI(m) = 1 - \frac{m^{(\text{def})} - m^{(\text{org})}}{m^{(\text{att})} - m^{(\text{org})}}. \quad (28)$$

An $RI(m)$ value closer to 1 indicates better robustness of the defense method.

5.3. Experimental Setup

We implemented the proposed method using the PyTorch version 2.2.0 deep learning framework. For the DSSM-Vis and VBPR, we used ResNet-50 as a feature extractor and a pretrained DDPM provided by ref. [42] for purifying the images. This DDPM is based on a U-Net architecture [46], which predicts the $\mu_\theta(x^t, t)$ and $\Sigma_\theta(x^t, t)$ for the denoising backward distribution at each time step. For the DSSM-Text, we used RoBERTa as a text encoder. The DDPM used in DSSM-Text is also based on a U-Net architecture; however, it is applied to the output of the RoBERTa encoder.

The denoising process was performed with $T_c = 36$, and the hyperparameters β_t were adjusted using a linear scheduling with $\beta_{\min} = 10^{-4}$ and $\beta_{\max} = 0.01$. We optimized the hyperparameters of the visual RSs (i.e., DSSM-Vis and VBPR) and DSSM-Text, as specified in ref. [10].

We preprocessed the images in HM and Amazon Men by resizing them to 224×224 and scaling their pixel values to the range $[0, 1]$. Moreover, we normalized the images to have a zero mean and a standard deviation of one before passing them to the baseline RSs. The images were rescaled to the range $[-1, +1]$ before being passed to the DDPM. We considered a maximum of 30 tokens for news titles in the MIND dataset, covering 99% of the total tokens.

We evaluated the performance of defense methods in both *standard* and *robust* settings. The standard setting evaluates the performance of defense methods on a clean, unaltered test set. On the other hand, the robust setting assesses their performance on an attacked test set. This shows how effectively defense methods can handle adversarial attacks.

5.4. Adversarial Attacks on Images

To estimate the effectiveness of the proposed method against adversarial attacks on images in visual RSs, we selected VBPR and DSSM-Vis as baseline RSs and evaluated our method against advanced threat models, including *Fast Gradient Sign Method* (FGSM), *Projected Gradient Descent* (PGD), and *Carlini & Wagner* (C&W). We also compared our method with SOTA defense methods, including *Adversarial Training* (AT) [11], *Free Adversarial Training* (FAT) [11], *Adversarial Image Denoiser* (AiD) [12], and *Adversarial Multimedia Recommendation* (AMR) [13].

FGSM is a white box attack in which the adversary has complete access to the baseline RS. It is a simple and efficient single-step attack, which generates adversarial examples by adding perturbations aligned with the loss gradient. In contrast, PGD is a stronger, iterative attack. It applies small, repeated perturbations over multiple steps. PGD also projects the perturbed input back into the allowed perturbation space. While PGD is more effective than FGSM, it is computationally more expensive due to its iterative nature. These attacks were performed with an l_∞ ball of radius = $8/255$ in our experiments.

The C&W is a powerful adversarial attack that generates adversarial examples using an optimization problem. It minimizes the perturbation required to mislead a model while ensuring the perturbed input remains close to the original.

Table 4 presents the performance of the competing defense methods against different attacks on the VBPR over the Amazon Men dataset. Additionally, Table 5 presents the performance of these methods versus different attacks on the DSSM-Vis over the HM dataset.

Table 4. Performance of the defense methods under different attacks on the VBPR over the Amazon Men dataset.

Attack	Method	HR@10	NDCG@10	HR@20	NDCG @20	RI(HR@10) %
PGD ($\epsilon = 8/255$)	No Defense	9.09	5.40	10.19	5.87	
	AT	14.85	9.43	17.64	9.50	68.74
	FAT	14.63	8.82	16.62	9.33	66.11
	AMR	14.17	9.01	18.75	10.33	60.62
	AiD	13.76	8.44	15.77	8.79	55.73
	UAPD	15.55	9.95	20.26	10.90	77.09
FGSM ($\epsilon = 8/255$)	No Defense	11.74	7.22	13.11	7.49	
	AT	15.02	9.64	18.29	10.06	57.24
	FAT	15.34	9.96	20.43	11.17	62.83
	AMR	14.78	9.46	19.33	10.31	53.05
	AiD	15.10	9.59	18.13	10.05	58.64
	UAPD	16.48	10.48	21.22	11.77	82.72
C&W	No Defense	10.19	6.44	13.01	7.03	
	AT	13.32	8.39	17.29	9.53	42.99
	FAT	15.28	9.76	18.90	10.41	69.92
	AMR	14.23	9.05	18.03	9.89	55.49
	AiD	13.02	8.39	16.25	8.93	38.87
	UAPD	16.36	10.49	20.8	11.41	84.75

The results highlight the effectiveness of various defense methods against different attacks in both experiments. For instance, under the PGD ($\epsilon = 8/255$) attack, the HR@20 and NDCG@20 of VBPR dropped to 10.19% and 5.87%, respectively. However, with Adversarial Training (AT), VBPR attained an HR@20 and NDCG@20 of 17.64% and 9.50%.

Additionally, the proposed method consistently outperformed other defense methods across different metrics. For example, under the PGD ($\epsilon = 8/255$) attack on the DSSM-Vis, UAPD achieved an HR@10 and NDCG@10 of 17.17% and 10.36%, respectively, while the second-best method, FAT, attained an HR@10 and NDCG@10 of 14.67% and 8.90%, respectively. Similarly, under the FGSM and C&W attacks, UAPD maintained superior performance, with the HR@20 reaching up to 22.76% and the NDCG@20 up to 12.22%.

Furthermore, we observe that UAPD's results across different attacks were almost stable, mainly due to the diffusion-based purification method in UAPD that can naturally eliminate any adversarial perturbation in the diffusion phase, irrespective of the attack type (refer to Section 4.4 for more details).

These findings show that the proposed DDPM training approach, guided denoising process, and detection mechanism can effectively remove adversarial perturbations generated by these advanced attacks. In the ablation study, we investigated the contribution of each mechanism to the overall performance of the proposed method.

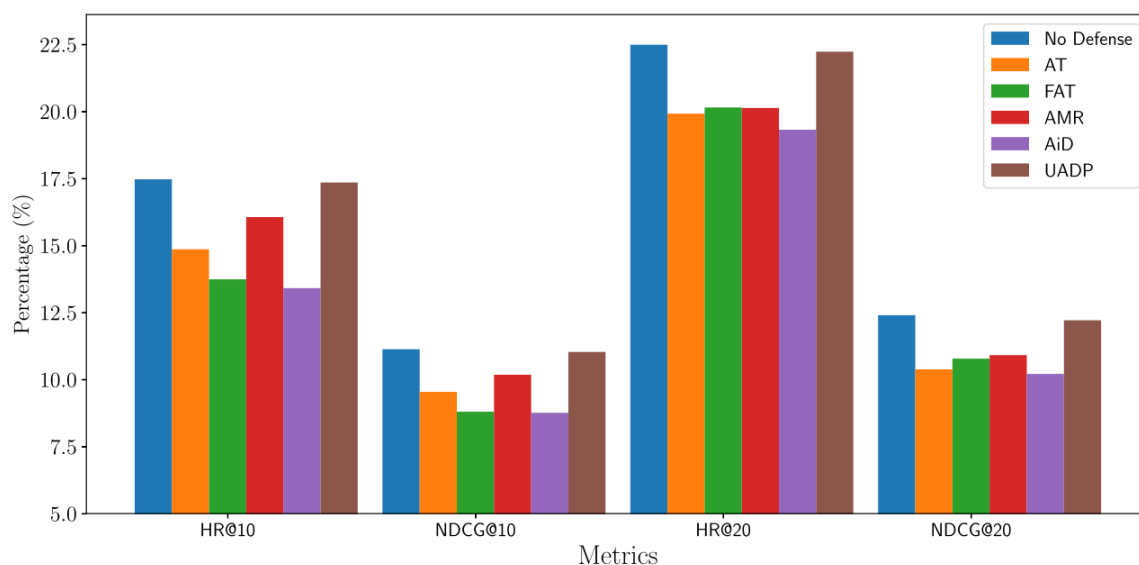
Table 5. Performance of the defense methods under different attacks on the DSSM-Vis over the HM dataset.

Attack	Method	HR@10	NDCG@10	HR@20	NDCG @20	RI(HR@10) %
PGD ($\epsilon = 8/255$)	No Defense	8.62	5.26	10.50	6.52	
	AT	11.97	7.32	18.03	9.57	34.22
	FAT	14.67	8.90	18.61	9.84	61.8
	AMR	14.39	8.88	19.43	10.42	58.94
	AiD	13.01	7.94	18.45	9.86	44.84
	UAPD	17.17	10.36	20.53	11.01	87.33
FGSM ($\epsilon = 8/255$)	No Defense	11.58	7.04	13.93	7.60	
	AT	16.32	9.90	20.69	11.14	69.4
	FAT	15.29	9.29	19.5	10.38	54.32
	AMR	14.28	8.71	18.28	9.78	39.53
	AiD	13.99	8.56	17.87	9.56	35.29
	UAPD	17.72	10.89	22.76	12.22	89.9
C&W	No Defense	10.96	6.68	13.22	7.35	
	AT	14.98	9.17	19.17	10.32	53.96
	FAT	14.71	8.98	18.80	10.07	50.34
	AMR	14.21	8.72	18.20	9.76	43.62
	AiD	14.57	9.02	18.61	10.17	48.46
	UAPD	17.03	10.47	21.86	11.65	81.48

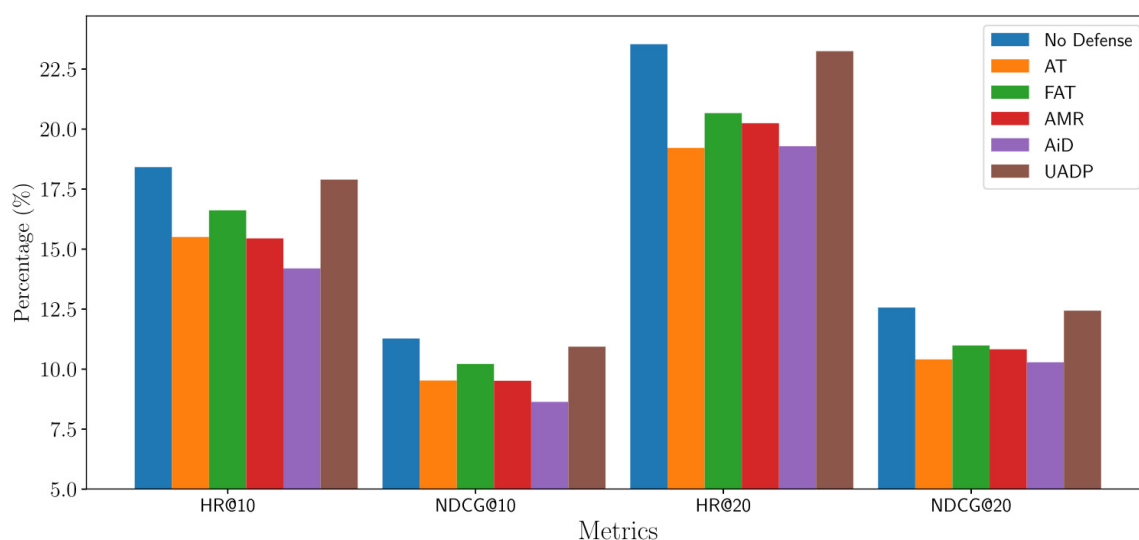
We also examined the standard performance of the defense methods on the baseline RSs. Figure 3 presents the results.

The results shed light on the standard performance—i.e., performance without any adversarial attacks—of different defense methods applied to the DSSM-Vis model on the HM dataset and the VBPR model on the Amazon Men dataset. The baseline models, DSSM-Vis and VBPR, exhibited the highest standard performance across all metrics. Specifically, DSSM-Vis achieved an HR@10 of 18.41% and an NDCG@10 of 11.27%, while VBPR achieved an HR@10 of 17.47% and an NDCG@10 of 11.13%.

The defense methods resulted in noticeable declines in both HR and NDCG metrics. For instance, the DSSM-Vis using AT achieved an HR@10 of 15.50% and an NDCG@10 of 9.52%, which was a significant drop from the baseline. Additionally, both baseline RSs using UAPD maintained performance levels much closer to the baseline. For instance, the DSSM-Vis using UAPD achieved an HR@10 of 17.89% and an NDCG@10 of 10.93%, which are only marginally lower than the baseline DSSM-Vis. Moreover, the VBPR using UAPD achieved an HR@10 of 17.35% and an NDCG@10 of 11.03%, nearly matching the baseline VBPR's performance. This minimal degradation indicates that UAPD effectively balances the trade-off between enhancing robustness against adversarial attacks and preserving standard recommendation accuracy.



(a) VBPR on the Amazon Men dataset.



(b) DSSM-Vis on the HM dataset.

Figure 3. Standard performance metrics (%) of the baseline RSs using the specified defense methods on the HM and Amazon Men datasets.

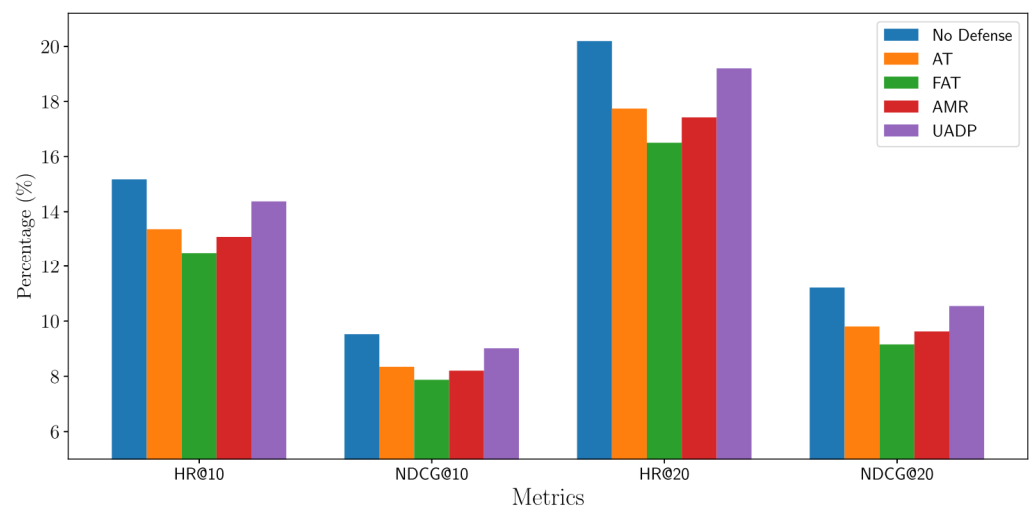
5.5. Adversarial Attacks on Textual Information

We also evaluated UAPD against adversarial attacks on the textual contents of an RS. To this end, we chose the DSSM-Text as the baseline RS. The performance of DSSM-Text with our defense method was assessed against PGD, FGSM, and C&W attacks. These attacks were applied to the output of the RoBERTa text encoder in the DSSM-Text. Additionally, we compared UAPD with peer defense methods including AT, FAT, and AMR. The perturbation size of PGD and FGSM was limited to $\epsilon = 0.2$ in our experiments. Table 6 presents the performance of the DSSM-Text using the competing defense methods on the MIND dataset. Additionally, Figure 4 compares the standard performance of the DSSM-Text using different defense methods.

The results show that all attacks successfully degraded the performance of the RS with no defense strategy. Among them, PGD was identified as the most effective threat model. For instance, without any defense, PGD decreased the HR@10 and NDCG@10 of the RS to 6.11% and 3.83%.

Table 6. Performance metrics (%) of the defense methods using the DSSM-Text as the baseline RS, under different attacks on the MIND dataset.

Attack	Method	HR@10	NDCG@10	HR@20	NDCG @20	RI(HR@10) %
PGD ($\epsilon = 0.2$)	No Defense	6.11	3.83	8.2	4.52	
	AT	12.01	7.54	15.92	8.84	65.12
	FAT	11.38	7.15	15.22	8.46	58.17
	AMR	11.02	6.91	14.58	8.14	54.19
	UAPD	13.04	8.19	17.4	9.69	76.49
FGSM ($\epsilon = 0.2$)	No Defense	8.31	5.16	11.02	6.20	
	AT	12.85	8.05	17.20	9.55	66.18
	FAT	11.48	7.13	15.28	8.51	46.21
	AMR	13.02	8.10	17.35	9.55	68.66
	UAPD	14.01	8.86	18.63	10.40	83.09
C&W	No Defense	7.36	4.65	9.87	5.34	
	AT	11.98	7.65	15.94	8.83	59.15
	FAT	11.23	7.08	14.91	8.35	49.55
	AMR	10.48	6.44	13.84	7.76	39.95
	UAPD	13.54	8.5	17.95	10.03	79.13

**Figure 4.** Standard performance metrics (%) of the DSSM-Text using the specified defense methods on the MIND dataset.

Additionally, all defense strategies substantially reduced the impact of the attack. For instance, the DSSM-Text with AT achieved an HR@10 of 12.01% under the PGD attack, far better than the 6.11% achieved by DSSM-Text with no defense method. However, the results show a noticeable drop compared to the standard performance of the DSSM-Text with an HR@10 of 15.17%.

Among the defense methods, UAPD consistently outperformed others across various metrics. For example, UAPD achieved an HR@10 and NDCG@10 of 13.54% and 8.5% under the C&W threat model, showing an improvement of 1.56% and 0.85% compared to AT, the second-best method, under the same attack.

As the results in Figure 4 indicate, all defense methods had side effects on the standard performance of the baseline RS. For instance, the NDCG@10 of the DSSM-Text decreased

from 9.52% to 7.87% when using the FAT. However, the proposed method consistently showed the minimum side effects among the defense methods.

The results of this experiment are consistent with previous findings regarding adversarial attacks on images. The results confirm that UAPD effectively balances the trade-off between enhancing robustness against adversarial attacks and preserving standard recommendation accuracy in the context of adversarial attacks in both image and textual modalities.

5.6. Shilling Attacks

This section presents experiments conducted to evaluate the performance of the proposed defense method against various shilling attacks. To this end, we considered random and bandwagon attacks [4], *Poisoning Recommender Systems* (PoisonRec) [14], *Deep learning* (DL)_Attack [15], and *Robust Adversarial Poisoning with Uncertainty* (RAPU) [16] attacks on two baseline RSs: the DSSM-Vis and the DSSM-Text. These attacks were implemented using the *Library for Attacks against Recommendation* (ARLib) [17]. We also compared our work with state-of-the-art defense methods, including *Adversarial Training* (AT) [47], *Adversarial Poisoning Training* (APT) [18], Bagging [19], and PORE [6].

We set the *fake user fraction* parameter to 5% for these attacks, the *target ratio* (the proportion of target items relative to the total item count) to 10%, and the *target item selection strategy* to “Unpopular”. The malicious rate size (the fraction of total items rated by fake users) was set to the average fraction of total items rated by real users. The hyperparameters of the competing methods were set according to their recommended settings. Table 7 presents the hyperparameters of the competing methods and their adjustment ranges.

Table 7. Hyperparameters of the defense methods and their adjustment ranges.

Hyperparameter	Method(s)	Description	Range
n_e	All	maximum number of epochs	{100}
d_e	All	embedding dimension	{64, 128}
lr	All	learning rate	$\{1 \times 10^{-5}, 5 \times 10^{-5}, 1 \times 10^{-4}\}$
bs	ALL	batch size	{64, 128}
ϵ	AT, APT	perturbation size	{0.03}
T_{pre}	APT	number of pre-training epochs	{5, 10, 15}
T_g	APT	adversarial users are generated every 2 to 3 epochs	{2, 3}
r^+	APT	the number of items rated positively by adversarial profiles	{1, 2}
n_{RS}	PORE, Bagging	number of recommender systems	100,000
s	PORE, Bagging	number of rows sampled from interaction matrix	{200, 500}
α	PORE	$1 - \alpha$ is a confidence score	0.001

Our experimental results showed that the specified shilling attacks were ineffective in terms of the global performance of the baseline RS. For example, the HR@10 and NDCG@10 of DSSM-Vis only decreased by 0.01% and 0.03% when applying the DL_Attack. Conversely, these attacks significantly manipulated the ranking of target items. For instance, the HR@10 and NDCG@10 of the target items increased by 25.22% and 239.06%, respectively, when we applied the DL_Attack. Therefore, we measured the performance of defense methods using the rate of increase in HR@k and NDCG@k of the target items ($k \in \{10, 20\}$), where a lower rate indicates a better defense.

Table 8 presents the performance of the DSSM-Vis using these defense methods under the specified attacks on the HM dataset. Additionally, Table 9 provides the results of the DSSM-Text using these defense methods on the MIND dataset.

As the results indicate, without any defense strategy, all attacks significantly enhanced the ranking of target items. Among these attacks, RAPU—formulated as a bi-level optimization problem—was the most effective threat model. For example, it increased the HR@10 and NDCG@10 of target items by 31.15% and 293.13% when the DSSM-Vis was used as the baseline RS and no defense method was adopted.

Table 8. Rate of increase (%) for target items using the DSSM-Vis as the baseline RS and the defense methods in the presence of the specified attacks on the HM dataset.

Attack	Method	HR@10	NDCG@10	HR@20	NDCG @20
Random	No Defense	22.53	207.90	30.84	208.77
	AT	2.69	24.77	3.68	24.92
	APT	2.75	25.19	3.76	25.33
	Bagging	2.49	22.56	3.36	22.66
	PORE	2.11	19.46	2.83	19.57
	UAPD	1.06	9.61	1.47	9.56
Bandwagon	No Defense	19.56	192.26	29.85	194.05
	AT	2.52	24.53	3.79	24.8
	APT	2.39	23.18	3.56	23.41
	Bagging	1.55	15.53	2.40	15.63
	PORE	1.96	19.01	2.91	19.07
	UAPD	1.61	15.78	2.48	15.95
PoisonRec	No Defense	26.54	244.61	37.66	275.58
	AT	3.12	30.36	4.74	30.63
	APT	3.27	31.94	4.96	32.29
	Bagging	3.43	34.04	5.32	34.45
	PORE	3.79	37.39	5.81	37.78
	UAPD	2.99	29.33	4.53	29.57
DL_Attack	No Defense	25.22	239.06	34.07	254.72
	AT	3.01	29.37	4.58	29.62
	APT	2.36	23.2	3.55	23.37
	Bagging	3.44	34.17	5.31	34.5
	PORE	3.9	38.75	6.02	39.08
	UAPD	2.41	24.2	3.73	24.47
RAPU	No Defense	31.15	293.13	41.92	298.53
	AT	3.37	33.02	5.14	33.30
	APT	2.77	26.94	4.12	27.18
	Bagging	3.16	30.95	4.83	31.22
	PORE	2.6	26.04	4.03	26.24
	UAPD	1.89	19.28	2.99	19.43

Table 9. Rate of increase (%) for target items using the DSSM-Text as the baseline RS and the defense methods in the presence of the specified attacks on the MIND dataset.

Attack	Method	HR@10	NDCG@10	HR@20	NDCG @20
Random	No defense	20.53	200.64	25.84	201.45
	AT	3.03	29.39	3.81	29.61
	APT	2.83	27.14	3.54	27.21
	Bagging	1.75	17.09	2.2	17.2
	PORE	0.55	5.09	0.69	5.06
	UAPD	1.3	12.59	1.62	12.58
Bandwagon	No defense	19.47	186.63	24.97	191.38
	AT	3.63	34.94	4.7	35.91
	APT	2.02	19.57	2.64	20.04
	Bagging	1.71	16.45	2.2	16.79
	PORE	1.98	18.64	2.51	19.06
	UAPD	0.85	8.10	1.11	8.2
PoisonRec	No defense	24.62	270.87	31.54	265.83
	AT	3.77	41.19	4.80	40.43
	APT	2.87	31.79	3.66	31.17
	Bagging	4.12	44.45	5.26	45.31
	PORE	4.02	43.53	5.17	44.35
	UAPD	1.86	20.37	2.35	20.02
DL_Attack	No defense	23.03	230.71	28.41	245.77
	AT	3.04	30.47	3.78	32.41
	APT	2.74	27.57	3.34	29.36
	Bagging	4.22	42.14	5.18	44.87
	PORE	5.12	51.56	6.35	54.88
	UAPD	2.89	28.98	3.59	30.88
RAPU	No defense	28.28	282.94	35.14	288.05
	AT	4.83	48.25	6.03	49.12
	APT	4.45	44.40	5.48	45.15
	Bagging	6.74	67.17	8.32	68.37
	PORE	5.44	54.25	6.75	55.24
	UAPD	3.56	35.76	4.47	36.34

Additionally, all defense methods substantially alleviated the effects of the specified attacks. Among them, UAPD was the best defense strategy against many of these attacks. For example, DSSM-Vis using UAPD achieved the best results in 3 of 5 attack scenarios. It reduced the increase rate of NDCG@10 for target items from 293.13% to 19.28% under the RAPU attack. Note that given the original NDCG@10 of target items was a low value of 0.0038, the remaining increase rate had a minimal impact in this case.

Among the adversarial methods (i.e., AT and APT), APT provided a better defense in most attack scenarios. For instance, DSSM-Text using APT decreased the increase rate of NDCG@10 for target items to 27.57% under the DL_Attack, whereas this rate decreased to 30.47% with AT. However, no defense strategy consistently outperformed the others across

all attacks, and the performance of the defense methods was competitive in many of these attack scenarios. Therefore, it seems that a combination of these defense methods could lead to a more robust and comprehensive defense.

5.7. Ablation Study

This experiment examined the contribution of the proposed weighting scheme and training process of the DDPM to the overall performance of the proposed method. To this end, we chose the DSSM-Vis as the baseline RS and derived two variants of the proposed method:

- UAPD w/o w: indicates UAPD without the weight associated with the input image by the DDPM;
- UAPD w/o train: refers to UAPD without the training process specified in Section 4.3.

We applied a PGD attack with an l_∞ ball of radius = 8/255 on the images from the HM dataset and measured the performance of these variants under this attack. Figure 5 illustrates the results.

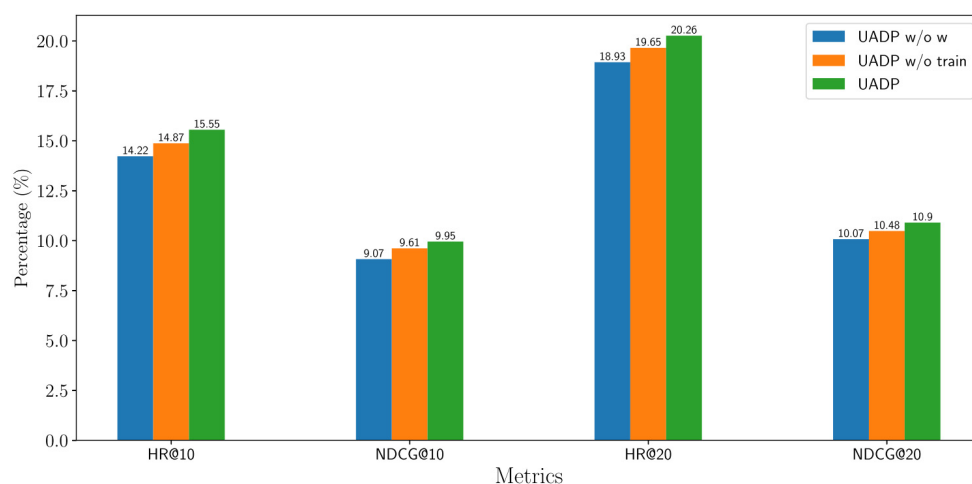


Figure 5. Ablation study of UAPD on the HM images under the PGD attack bounded by an l_∞ ball of radius = 8/255.

The results show that both the weighting scheme and the training process of the DDPM were effective and positively impacted the overall performance of UAPD. The weighting scheme had a greater positive impact compared to the training process. This was mainly due to the purification power of the pretrained DDPM, which could effectively eliminate adversarial noise without a fine-tuning step. However, the training process is necessary for other modalities like text. Moreover, this process noticeably boosts the purification power of the DDPM on images.

5.8. Hyperparameter Analysis

This section explores two hyperparameters of the proposed method: T_c and λ_{EMA} . The parameter T_c determines the number of purification steps, whereas the parameter λ_{EMA} controls the update rate of interaction labels in the self-adaptive process.

In the first experiment, we chose the DSSM-Vis as the baseline RS, randomly selected 50% of the HM's images, and then applied a PGD attack with an l_∞ norm ball of $\epsilon = 8/255$. Then, we measured the RI(HR@10) and RI(NDCG@10) of UAPD by varying the value of T_c in {20, 30, 36, 40, 50, 60, 70, 100}. The results are illustrated in Figure 6.

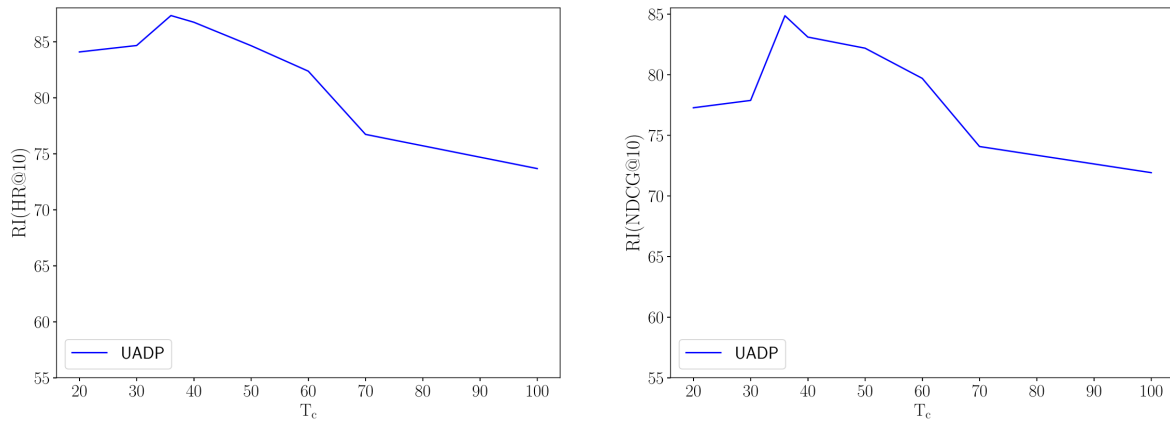


Figure 6. RI(HR@10) and RI(NDCG@10) of UADP by varying purification steps (T_c) under the PGD attack with l_∞ -norm ball of $\epsilon = 8/255$ on the HM dataset.

The results indicate that as the value of T_c increases, the effectiveness of the purification improves, reaching peak performance in the range $T_c \in (30, 40)$. In this range, UADP achieved a RI(HR@10) over 85%, indicating strong resistance to the attack. Afterward, the performance of the proposed method smoothly declined, mainly due to the side effects of excessive purification of the clean images.

In the second experiment, we applied a RAPU attack on the DSSM-Vis using UADP as the defense method with the following parameters: *fake user fraction* = 5%, *target ratio* = 10%, and *target item selection strategy* = “Unpopular.” The malicious rate size was also set to the average fraction of total items rated by real users. Then, we measured the RI(HR@10) and RI(NDCG@10) of the *target items* by varying the value of λ_{EMA} in {0.6, 0.7, 0.8, 0.9, 0.95, 0.99}. Figure 7 shows the results.

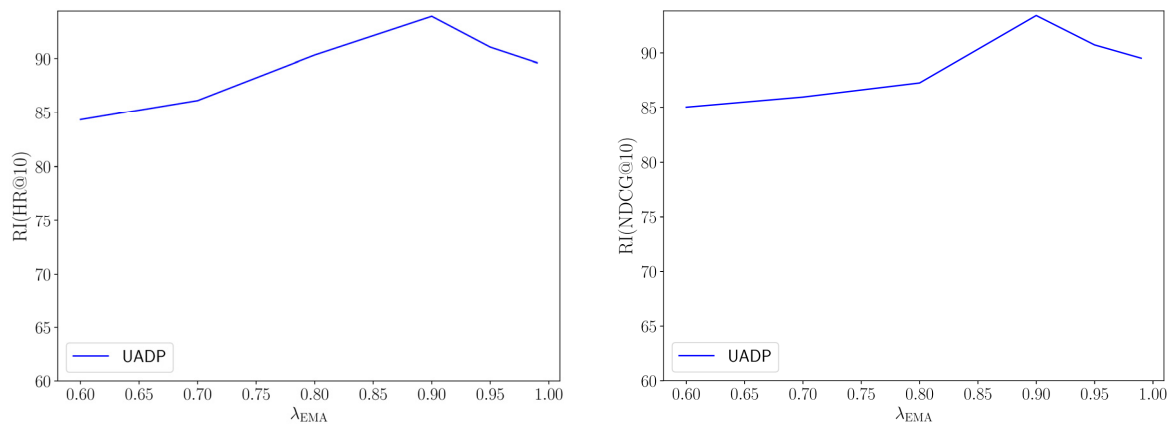


Figure 7. RI(HR@10) and RI(NDCG@10) of UADP by varying λ_{EMA} under the RAPU attack on the HM dataset.

The outcomes reveal that the performance of UADP against this attack gradually improved as the value of λ_{EMA} increased from 0.60 to 0.90. The peak performance was observed at $\lambda_{EMA} = 0.90$. At this point, UADP achieved an RI(NDCG@10) of 93.42%. Afterward, the performance of UADP gradually declined. Moreover, the high performance of UADP in the wide range $\lambda_{EMA} \in (0.85, 0.99)$ indicates the low sensitivity of our method to specific values of this hyperparameter.

5.9. Robustness

To evaluate the robustness of the proposed method, we applied PGD attacks with different perturbation sizes (ϵ) to the HM’s images and assessed the decline in the performance

of the DSSM-Vis with UAPD. Similarly, adversarial perturbations of varying magnitudes were applied to the MIND's item descriptions and the decline in the performance of the DSSM-Text with UAPD was measured.

A smaller decrease indicates greater robustness. Figure 8 compares the robustness of UAPD with that of AMR on the DSSM-Vis and DSSM-Text, under various levels of adversarial perturbation.

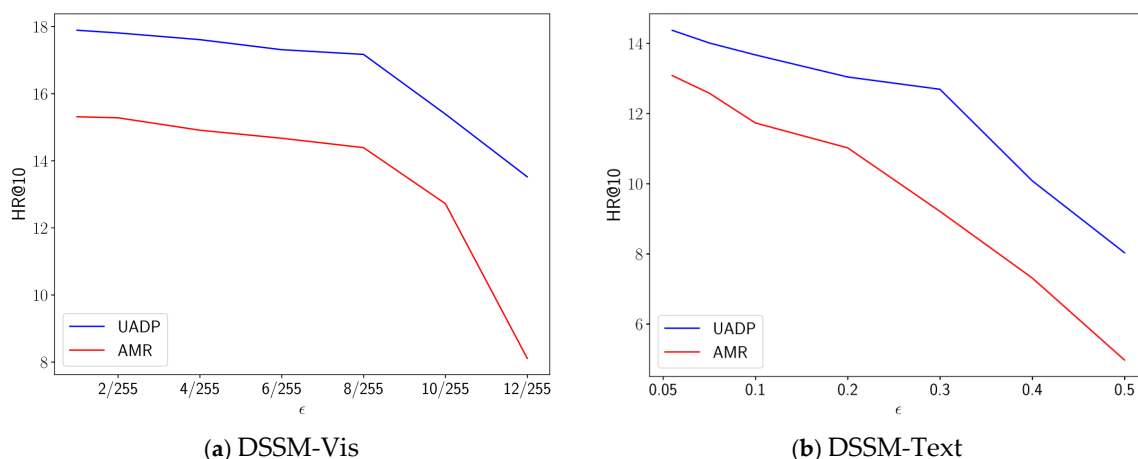


Figure 8. Robustness of the baseline RSs under the PGD attack by varying ϵ (perturbation size).

We can observe that for different values of ϵ , UAPD outperformed AMR by a relatively large margin when DSSM-Vis was selected as the baseline RS. Moreover, the DSSM-Vis achieved strong robustness under this attack using both defense methods when $\epsilon \leq 8/255$. For greater perturbation sizes, which may become noticeable to the human eye, there were considerable performance drops. However, UAPD had a much smaller performance drop than AMR, highlighting its superior robustness under more intense attack scenarios.

The results on the text modality were consistent with the outcomes of adversarial attacks on images. UAPD consistently provided higher performance for the DSSM-Text compared to AMR. For perturbation sizes exceeding 0.2, both defense methods experienced a substantial performance decline. However, UAPD showed a significantly smaller performance decline compared to AMR.

The results on both modalities provide empirical evidence of the robustness of UAPD. UAPD is less vulnerable to adversarial attacks compared to Adversarial Training methods because of the purification capabilities of the DDPM and the weighting scheme in the proposed method.

In the second experiment, we evaluated the robustness of UAPD against shilling attacks. To this end, we chose the DSSM-Vis as the baseline and applied the RAPU attack with the following parameters: *target ratio* = 10% and *target item selection strategy* = “Unpopular.” The number of ratings provided by malicious users was set to match the average fraction of total items that genuine users rate. Then, we measured the RI(HR@10) and RI(NDCG@10) of *target items* by varying the value of the *fake user fraction* (η) in {5%, 7%, 10%, 12%, 15%, 20%}. Figure 9 compares the results of UAPD with those of APT.

The results indicate that for realistic values of (i.e., $\eta \leq 10\%$), UAPD consistently maintains high robustness against the RAPU attack. In this range, UAPD shows minimal variance in performance metrics, maintaining an RI(HR@10) of over 90%, which suggests that UAPD effectively mitigates the impact of adversarial users under plausible attack scenarios. For larger values of η , which may be unrealistic in a real-world situation, the RI(HR@10) of UAPD drops slightly. For instance, the RI(HR@10) of UAPD declines to 79.41% when the value of η increases to 18%. In contrast, APT experiences a significantly

larger performance drop compared to our method, as its RI(HR@10) declines to 68.15% for $\eta = 18\%$. Thus, we can conclude that UAPD is robust against advanced shilling attacks such as RAPU, especially when the *fake user fraction* is limited to 10%.

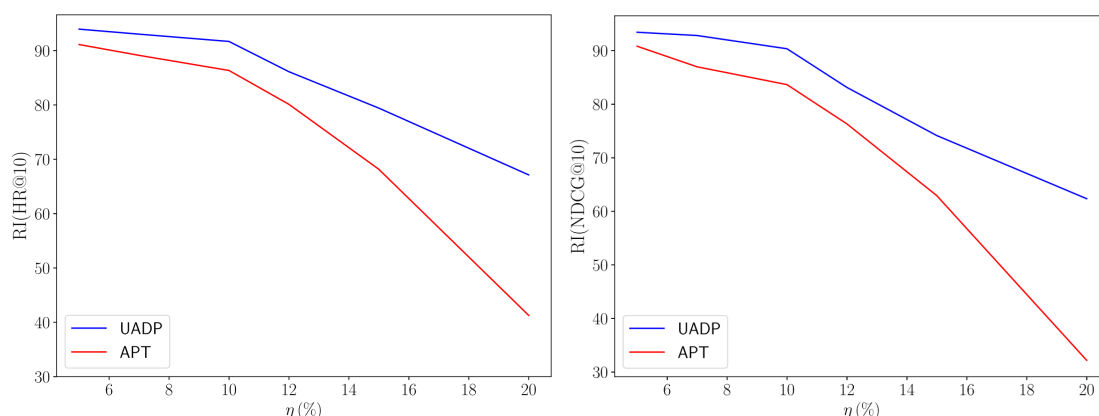


Figure 9. Robustness of the DSSM-Vis under the RAPU attack by varying η (fake user fraction).

We also evaluated the performance of UAPD against simultaneous attacks on both the inputs and interaction matrix of the DSSM-Vis. To this end, we simultaneously applied the PGD attack to 50% of the HM's input images and the RAPU attack to the interaction matrix of the dataset. The PGD attack reduces the global performance of the RS, whereas the RAPU attack mainly affects the ranking metrics of the target items. Here, we set $T_c = 36$ and $\lambda_{EMA} = 0.9$ and evaluated the *global* HR@10 of the RS, along with the RI(HR@10) of the *target items* by varying the intensity of the attacks. Specifically, we measured the HR@10 of UAPD by varying the value of ϵ parameter of the PGD attack in the range $\{2/255, 4/255, 8/255, 10/255\}$ and the η parameter of the RAPU attack within the range $\{5\%, 10\%, 15\%\}$. Moreover, we evaluated the RI(HR@10) of the *target items* by varying the value of η in the range $\{5\%, 7\%, 10\%, 12\%, 15\%, 20\%\}$ and ϵ within the range $\{4/255, 8/255, 10/255\}$. Figure 10 illustrates the results.

The results show that for realistic values of ϵ and (i.e., $\epsilon \leq 8/255$ and $\eta \leq 10\%$), the global HR@10 of DSSM-Vis using UAPD as the defense method is largely preserved under the combined attack. In this range, the performance of UAPD under the combined attack is comparable to its performance under the PGD attack alone. For instance, the HR@10 of the DSSM-Vis slightly declines from 17.17% to 17.09% when $\epsilon = 8/255$ and η increases from 0% to 10%. For larger values of η , the RAPU attack influences the performance of the RS to a greater extent; however, the performance remains within an acceptable range even in this extreme scenario. For example, applying the RAPU attack with $\eta = 15\%$ reduces the HR@10 of the RS from 17.17% to 16.81%. Only at extreme values of $\epsilon = 12/255$ and $\eta = 15\%$ do we observe a considerable performance drop. Therefore, we can conclude that UAPD maintains the performance of the RS under simultaneous attacks, particularly for realistic attack sizes.

Additionally, we observe that for realistic values of ϵ and (i.e., $\epsilon \leq 8/255$ and $\eta \leq 10\%$), the RI(HR@10) of target items remains largely preserved. For instance, the RI(HR@10) of target items under only the RAPU ($\eta = 5\%$) attack is 93.94%. When the PGD ($\epsilon = 8/255$) attack is applied simultaneously with the RAPU attack, the RI(HR@10) of target items declines slightly to 92.34%. However, the RI(HR@10) of target items shows a greater decline for the extreme value of (i.e., $\epsilon = 10/255$) and $\eta > 10\%$. By simultaneously applying the PGD ($\epsilon = 10/255$) and the RAPU ($\eta = 20\%$) attacks, the RI(HR@10) of target items is significantly dropped and reaches 58.65%. Thus, we can conclude that UAPD is robust against targeted shilling attacks even when it simultaneously faces a strong adversarial

attack like PGD. However, a combined attack of extreme sizes can significantly weaken this robustness.

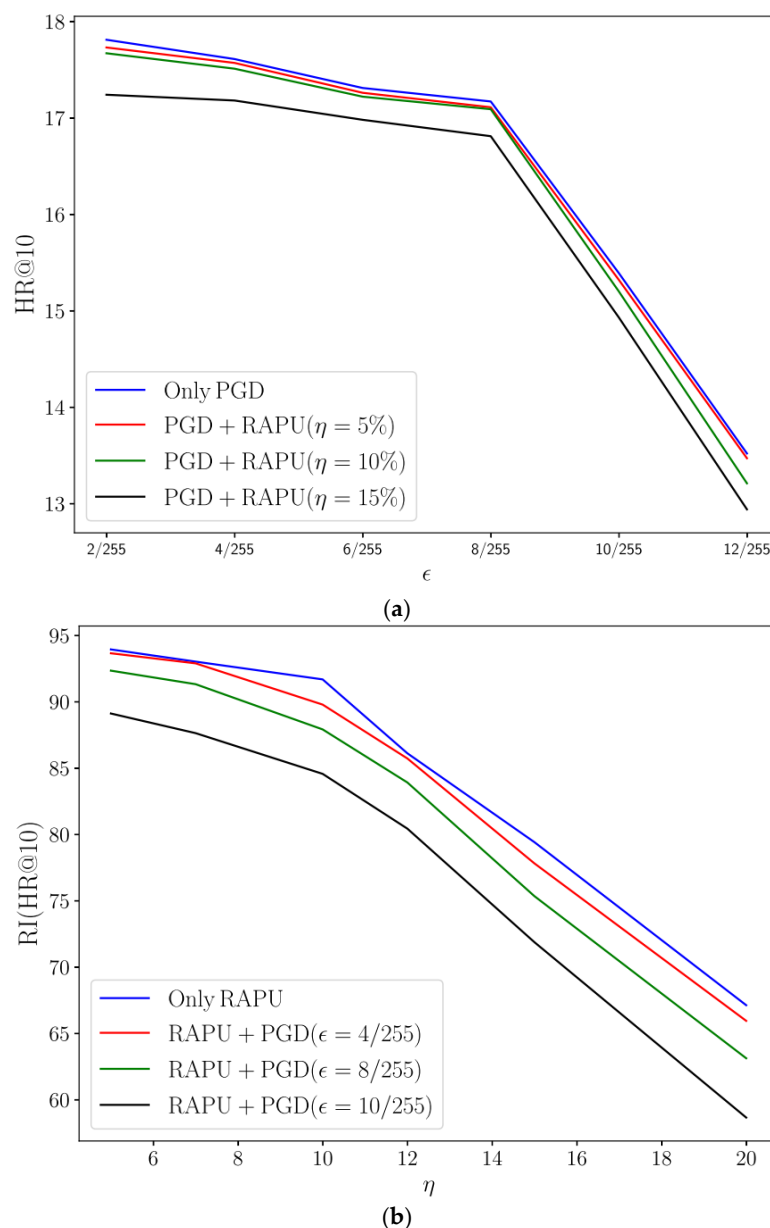


Figure 10. Robustness of the DSSM-Vis under simultaneous shilling and adversarial attacks of varying sizes. (a) HR@10 of DSSM-Vis using UAPD as the defense method under simultaneous attacks of varying sizes. (b) RI(HR@10) of target items using UAPD as the defense method under simultaneous attacks of varying sizes.

6. Conclusions and Future Work

This research introduces UAPD, a unified input/target attack purifier and detector for RSs. UAPD enhances the purification power of diffusion networks through a novel training procedure. It also presents an effective adversarial detection mechanism using the output of diffusion networks. Additionally, UAPD adapts the self-adaptive training strategy for RSs, enabling our model to detect and refine the interaction matrix of a baseline RS during the training process by leveraging its own predictions. Moreover, it introduces a novel metric for detecting fake users by utilizing the output of the target refinement procedure.

We conducted extensive experiments on three large-scale RS benchmarks and compared UAPD's performance with several leading defense methods. The outcomes confirm

that the proposed method consistently outperforms other defense methods under advanced adversarial attacks on both images and textual content in RSs.

Additionally, the results against several shilling attacks reveal that UAPD, by achieving the best results in three out of five attack scenarios, is the best defense strategy among the competing methods.

The results of the ablation study indicate that the DDPM training process and the proposed weighting scheme improve UAPD's performance. We also evaluated the robustness of UAPD by measuring its performance against attacks of varying intensities on both the content and the interaction matrix of a baseline RS. The results show that for attacks with realistic magnitudes, UAPD maintains the performance of the baseline RS even under simultaneous attacks.

In future work, we aim to extend our defense method to other information sources of RSs, such as social connections. Furthermore, evaluating the effectiveness of UAPD on state-of-the-art (SOTA) sequential RS models, such as refs. [48–50], represents another promising research direction. We will also explore the effectiveness of combining multiple defense methods against shilling attacks.

Author Contributions: Conceptualization, D.Z. and A.A.; Formal analysis, A.A. and D.Z.; Investigation, S.A. and D.Z.; Methodology, D.Z. and A.A.; Software, A.A.; Supervision, S.A. and D.Z.; Writing—original draft, A.A.; Writing—review and editing, S.A. and D.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Datasets used in the experiments are publicly available and can be downloaded from the following links: HM and MIND: <https://github.com/westlake-repl/IDvs.MoRec> (accessed on 2 January 2025). Amazon Men: <http://jmcauley.ucsd.edu/data/amazon> (accessed on 2 January 2025).

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Proof of Equation (15)

For brevity, let $\mu = \mu_\theta(x^{(t+1)}, t)$ and $\Sigma = \Sigma_\theta(x^{(t+1)}, t)$. Let also $\{C_i | i = 1, 2, \dots, 5\}$ represent a set of normalization constants. Using the Bayes' rule, we have the following:

$$p_\theta(x^t | x^{(t+1)}, x^{0t}) \propto p(x^{0t} | x^t, x^{(t+1)}) p_\theta(x^t | x^{(t+1)})$$

Now, assume x^{0t} by knowing x^t is independent of $x^{(t+1)}$, which gives

$$p_\theta(x^t | x^{(t+1)}, x^{0t}) \propto p(x^{0t} | x^t) p_\theta(x^t | x^{(t+1)}) \quad (A1)$$

Therefore

$$\ln p_\theta(x^t | x^{(t+1)}, x^{0t}) = \ln p(x^{0t} | x^t) + \ln p_\theta(x^t | x^{(t+1)}) + C_1 \quad (A2)$$

Since $p_\theta(x^t | x^{(t+1)}) = \mathcal{N}(x^t | \mu, \Sigma)$, we have

$$\ln p_\theta(x^t | x^{(t+1)}) = -\frac{1}{2}(x^t - \mu)^\top \Sigma^{-1}(x^t - \mu) + C_2 \quad (A3)$$

Now, let us assume that $p(x^{0t}|x^t)$ exhibits relatively low curvature compared to Σ^{-1} . Under this assumption, $p(x^{0t}|x^t)$ can be approximated using a Taylor series expansion around $x^t = \mu$ as:

$$\begin{aligned}\ln p(x^{0t}|x^t) &\approx (x^t - \mu) \nabla_{x^t} \ln p(x^{0t}|x^t) \Big|_{x^t=\mu} + \ln p(x^{0t}|x^t) \Big|_{x^t=\mu} \\ &= (x^t - \mu)g + C_3,\end{aligned}\quad (A4)$$

where $g = \nabla_{x^t} \ln p(x^{0t}|x^t) \Big|_{x^t=\mu}$. Substituting (d) and (c) in (b) yields

$$\begin{aligned}\ln p_\theta(x^t|x^{(t+1)}, x^{0t}) &\approx -\frac{1}{2}(x^t - \mu)^\top \Sigma^{-1}(x^t - \mu) + (x^t - \mu)g + C_4 \\ &= -\frac{1}{2}(x^t - \mu - \Sigma g)^\top \Sigma^{-1}(x^t - \mu - \Sigma g) + \frac{1}{2}g^\top \Sigma g + C_4 \\ &= -\frac{1}{2}(x^t - \mu - \Sigma g)^\top \Sigma^{-1}(x^t - \mu - \Sigma g) + C_5\end{aligned}\quad (A5)$$

Therefore, we can conclude that

$$p_\theta(x^t|x^{(t+1)}, x^{0t}) \approx \mathcal{N}(x^t|\mu + \Sigma g, \Sigma),$$

which completes the proof.

References

- Ovaisi, Z.; Heinecke, S.; Li, J.; Zhang, Y.; Zheleva, E.; Xiong, C. Rgrecsys: A toolkit for robustness evaluation of recommender systems. In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, Virtual, 21–25 February 2022; pp. 1597–1600.
- Calandrino, J.A.; Kilzer, A.; Narayanan, A.; Felten, E.W.; Shmatikov, V. “You might also like:” Privacy risks of collaborative filtering. In Proceedings of the 2011 IEEE Symposium on Security and Privacy, Oakland, CA, USA, 22–25 May 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 231–246.
- Fang, M.; Yang, G.; Gong, N.Z.; Liu, J. Poisoning attacks to graph-based recommender systems. In Proceedings of the 34th Annual Computer Security Applications Conference, San Juan, PR, USA, 3–7 December 2018; pp. 381–392.
- Gunes, I.; Kaleli, C.; Bilge, A.; Polat, H. Shilling attacks against recommender systems: A comprehensive survey. *Artif. Intell. Rev.* **2014**, *42*, 767–799. [\[CrossRef\]](#)
- Ge, Y.; Liu, S.; Fu, Z.; Tan, J.; Li, Z.; Xu, S.; Li, Y.; Xian, Y.; Zhang, Y. A survey on trustworthy recommender systems. *ACM Trans. Recomm. Syst.* **2022**, *3*, 1–68.
- Jia, J.; Liu, Y.; Hu, Y.; Gong, N.Z. PORE: Provably Robust Recommender Systems against Data Poisoning Attacks. *arXiv* **2023**, arXiv:2303.14601.
- Wang, X.; Cai, Y.; Li, F. Finite-time decentralized event-triggered state estimation for coupled neural networks under unreliable Markovian network against mixed cyberattacks. *Chin. Phys. B* **2024**, *33*, 110207. [\[CrossRef\]](#)
- Huang, L.; Zhang, C.; Zhang, H. Self-adaptive training: Bridging supervised and self-supervised learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *46*, 1362–1377. [\[CrossRef\]](#) [\[PubMed\]](#)
- He, R.; McAuley, J. VBPR: Visual bayesian personalized ranking from implicit feedback. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; Volume 30.
- Yuan, Z.; Yuan, F.; Song, Y.; Li, Y.; Fu, J.; Yang, F.; Pan, Y.; Ni, Y. Where to go next for recommender systems? id-vs. modality-based recommender models revisited. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, Taipei, Taiwan, 23–27 July 2023; pp. 2639–2649.
- Anelli, V.W.; Deldjoo, Y.; Di Noia, T.; Malitesta, D.; Merra, F.A. A study of defensive methods to protect visual recommendation against adversarial manipulation of images. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Montreal, QC, Canada, 11–15 July 2021; pp. 1094–1103.
- Merra, F.A.; Anelli, V.W.; Di Noia, T.; Malitesta, D.; Mancino, A.C.M. Denoise to Protect: A Method to Robustify Visual Recommenders from Adversaries. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, Taipei, Taiwan, 23–27 July 2023; pp. 1924–1928.
- Tang, J.; Du, X.; He, X.; Yuan, F.; Tian, Q.; Chua, T.-S. Adversarial training towards robust multimedia recommender system. *IEEE Trans. Knowl. Data Eng.* **2019**, *32*, 855–867. [\[CrossRef\]](#)

14. Song, J.; Li, Z.; Hu, Z.; Wu, Y.; Li, Z.; Li, J.; Gao, J. Poisonrec: An adaptive data poisoning framework for attacking black-box recommender systems. In Proceedings of the 2020 IEEE 36th International Conference on Data Engineering (ICDE), Dallas, TX, USA, 20–24 April 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 157–168.
15. Huang, H.; Mu, J.; Gong, N.Z.; Li, Q.; Liu, B.; Xu, M. Data poisoning attacks to deep learning based recommender systems. *arXiv* **2021**, arXiv:2101.02644.
16. Zhang, H.; Tian, C.; Li, Y.; Su, L.; Yang, N.; Zhao, W.X.; Gao, J. Data poisoning attack against recommender system using incomplete and perturbed data. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Virtual, 14–18 August 2021; pp. 2154–2164.
17. Wang, Z.; Gao, M.; Yu, J.; Ma, H.; Yin, H.; Sadiq, S. Poisoning attacks against recommender systems: A survey. *arXiv* **2024**, arXiv:2401.01527.
18. Wu, C.; Lian, D.; Ge, Y.; Zhu, Z.; Chen, E.; Yuan, S. Fight fire with fire: Towards robust recommender systems via adversarial poisoning training. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Montreal, QC, Canada, 11–15 July 2021; pp. 1074–1083.
19. Jia, J.; Cao, X.; Gong, N.Z. Intrinsic certified robustness of bagging against data poisoning attacks. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; Volume 35, pp. 7961–7969.
20. Lam, S.K.; Riedl, J. Shilling recommender systems for fun and profit. In Proceedings of the 13th International Conference on World Wide Web, New York, NY, USA, 17–20 May 2004; pp. 393–402.
21. O’mahony, M.P.; Hurley, N.J.; Silvestre, G.C. An evaluation of neighbourhood formation on the performance of collaborative filtering. *Artif. Intell. Rev.* **2004**, *21*, 215–228. [[CrossRef](#)]
22. Mobasher, B.; Burke, R.; Bhaumik, R.; Williams, C. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Trans. Internet Technol. (TOIT)* **2007**, *7*, 23–es. [[CrossRef](#)]
23. Yang, Z.; Xu, L.; Cai, Z.; Xu, Z. Re-scale AdaBoost for attack detection in collaborative filtering recommender systems. *Knowl.-Based Syst.* **2016**, *100*, 74–88. [[CrossRef](#)]
24. Fang, M.; Gong, N.Z.; Liu, J. Influence function based data poisoning attacks to top-n recommender systems. In Proceedings of the Web Conference 2020, Taipei, Taiwan, 20–24 April 2020; pp. 3019–3025.
25. Hu, R.; Guo, Y.; Pan, M.; Gong, Y. Targeted poisoning attacks on social recommender systems. In Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM), Waikoloa, HI, USA, 9–13 December 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6.
26. Bhaumik, R.; Mobasher, B.; Burke, R. A Clustering Approach to Unsupervised Attack Detection in Collaborative Recommender Systems. In Proceedings of the International Conference on Data Science (ICDATA), 2011; Citeseer; p. 1. Available online: https://librarysearch.adelaide.edu.au/discovery/fulldisplay/alma9928264932801811/61ADELAIDE_INST:UOFA (accessed on 7 January 2025).
27. Zhang, Y.; Tan, Y.; Zhang, M.; Liu, Y.; Chua, T.-S.; Ma, S. Catch the black sheep: Unified framework for shilling attack detection based on fraudulent action propagation. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015.
28. Zhang, F.; Lu, Y.; Chen, J.; Liu, S.; Ling, Z. Robust collaborative filtering based on non-negative matrix factorization and L1-norm. *Knowl.-Based Syst.* **2017**, *118*, 177–190. [[CrossRef](#)]
29. Yu, H.; Gao, R.; Wang, K.; Zhang, F. A novel robust recommendation method based on kernel matrix factorization. *J. Intell. Fuzzy Syst.* **2017**, *32*, 2101–2109. [[CrossRef](#)]
30. Ebrahimian, M.; Kashef, R. Detecting shilling attacks using hybrid deep learning models. *Symmetry* **2020**, *12*, 1805. [[CrossRef](#)]
31. Si, M.; Li, Q. Shilling attacks against collaborative recommender systems: A review. *Artif. Intell. Rev.* **2020**, *53*, 291–319. [[CrossRef](#)]
32. Backstrom, L.; Leskovec, J. Supervised random walks: Predicting and recommending links in social networks. In Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, Hong Kong, China, 9–12 February 2011; pp. 635–644.
33. Rezaimehr, F.; Dadkhah, C. T&TRS: Robust collaborative filtering recommender systems against attacks. *Multimed. Tools Appl.* **2024**, *83*, 31701–31731.
34. Zhang, K.; Cao, Q.; Wu, Y.; Sun, F.; Shen, H.; Cheng, X. Lorec: Large language model for robust sequential recommendation against poisoning attacks. *arXiv* **2024**, arXiv:2401.17723.
35. Wang, W.; Xu, Y.; Feng, F.; Lin, X.; He, X.; Chua, T.-S. Diffusion recommender model. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, Taipei, Taiwan, 23–27 July 2023; pp. 832–841.
36. Zhao, J.; Wang, W.; Xu, Y.; Sun, T.; Feng, F. Plug-In Diffusion Model for Embedding Denoising in Recommendation System. *CoRR* **2024**, arXiv:2401.06982.
37. Gao, J.; Lanchantin, J.; Soffa, M.L.; Qi, Y. Black-box generation of adversarial text sequences to evade deep learning classifiers. In Proceedings of the 2018 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 24 May 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 50–56.

38. Di Noia, T.; Malitesta, D.; Merra, F.A. Taamr: Targeted adversarial attack against multimedia recommender systems. In Proceedings of the 2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W), Valencia, Spain, 29 June–2 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–8.
39. Carlini, N.; Wagner, D. Audio adversarial examples: Targeted attacks on speech-to-text. In Proceedings of the 2018 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 24 May 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–7.
40. Liu, F.; Chen, H.; Cheng, Z.; Liu, A.; Nie, L.; Kankanhalli, M. Disentangled multimodal representation learning for recommendation. *IEEE Trans. Multimed.* **2022**, *25*, 7149–7159. [[CrossRef](#)]
41. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv* **2014**, arXiv:1412.6572.
42. Dhariwal, P.; Nichol, A. Diffusion models beat gans on image synthesis. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 8780–8794.
43. Wu, Q.; Ye, H.; Gu, Y. Guided diffusion model for adversarial purification from random noise. *arXiv* **2022**, arXiv:2206.10875.
44. Rendle, S.; Freudenthaler, C.; Gantner, Z.; Schmidt-Thieme, L. BPR: Bayesian personalized ranking from implicit feedback. *arXiv* **2012**, arXiv:1205.2618.
45. Wu, F.; Qiao, Y.; Chen, J.-H.; Wu, C.; Qi, T.; Lian, J.; Liu, D.; Xie, X.; Gao, J.; Wu, W. Mind: A large-scale dataset for news recommendation. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020; pp. 3597–3606.
46. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015: 18th International Conference, Munich, Germany, 5–9 October 2015, Proceedings, Part III* 18; Springer: Berlin/Heidelberg, Germany, 2015; pp. 234–241.
47. He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; Chua, T.-S. Neural collaborative filtering. In Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 3–7 April 2017; pp. 173–182.
48. Luo, J.; Gu, Y.; Luo, X.; Ju, W.; Xiao, Z.; Zhao, Y.; Yuan, J.; Zhang, M. GALA: Graph Diffusion-based Alignment with Jigsaw for Source-free Domain Adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 9038–9051. [[CrossRef](#)]
49. Qin, Y.; Wu, H.; Ju, W.; Luo, X.; Zhang, M. A diffusion model for poi recommendation. *ACM Trans. Inf. Syst.* **2023**, *42*, 1–27. [[CrossRef](#)]
50. Ju, W.; Qin, Y.; Qiao, Z.; Luo, X.; Wang, Y.; Fu, Y.; Zhang, M. Kernel-based substructure exploration for next POI recommendation. In Proceedings of the 2022 IEEE International Conference on Data Mining (ICDM), Orlando, FL, USA, 28 November–1 December 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 221–230.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.