Hanoi University of Science and Technology

School of Information and Communication Technology

_____ * _____

Data Entrepreneurship in Action

Topic: **Adversarial Personalized Ranking for Recommendation**

Supervisor: **Dr. Nguyen Kiem Hieu**

Hanoi, April 07, 2024

Contents

1 Introduction

In today's digital age, where the abundance of content and products can overwhelm users, recommendation systems play a key role in guiding decision-making and enhancing satisfaction of the user. These systems leverage machine learning algorithms to analyze user preferences and behavior, ultimately providing recommendations tailored to each individual's preferences and needs.

From movie recommendations on Netflix to product recommendations on Amazon and video curation on YouTube, recommender systems have become ubiquitous across online platforms, reshaping the way users interact with content and services digitally. By harnessing vast amounts of data, these systems can accurately predict items that are likely to resonate with users, thereby streamlining the browsing and discovery process.

Recommender systems, while invaluable in enhancing user experience and engagement, are not immune to various forms of attacks aimed at manipulating recommendations, compromising user privacy, or undermining the integrity of the system. Understanding these potential attacks is crucial for developing robust defenses and ensuring the reliability and trustworthiness of recommender systems. In our work, we just focus on "adversarial attack and defense" on recommender systems.

Adversarial attacks involve intentionally manipulating input data to deceive or disrupt the performance of machine learning models, including recommender systems. Here's an overview of attack strategies against recommender systems:

Profile Poisoning: In profile poisoning attacks, adversaries manipulate user profiles or feedback data to introduce bias into the recommender system. By injecting fake ratings, preferences, or interactions, attackers aim to distort the system's understanding of user preferences, leading to inaccurate recommendations.

Data Poisoning Similar to profile poisoning, data poisoning attacks involve injecting malicious data into the training dataset used by the recommender system. This can include adding fake items or altering existing data points to manipulate the model's behavior. As a result, the recommender system may learn incorrect patterns or associations, leading to compromised recommendations.

Model Evasion: In model evasion attacks, adversaries exploit vulnerabilities in the recommendation algorithm to manipulate the recommendation process. This can involve crafting input data, such as queries or user profiles, to force the system to produce specific recommendations or to suppress certain items from being recommended.

Model Inversion: Model inversion attacks involve inferring sensitive information about users or items by exploiting the outputs of the recommender system. By analyzing the recommendations made by the system, attackers may reverse-engineer underlying user preferences or characteristics, potentially compromising user privacy.

Data Poisoning with Malicious Items: Adversaries may introduce malicious items into the recommender system to influence recommendations in undesirable ways. These malicious items can be designed to manipulate user preferences, promote certain products, or even deceive users.

Sybil Attacks: In Sybil attacks, adversaries create multiple fake user accounts or profiles to influence the recommendations generated by the system. By controlling a network of Sybil identities, attackers can artificially inflate the perceived popularity or relevance of certain items, thereby biasing the recommendations.

These attack strategies highlight the vulnerabilities that recommender systems may face in real-world scenarios. Addressing these challenges requires the development of robust defense mechanisms, such as anomaly detection, adversarial training, and input validation, to mitigate the impact of adversarial attacks and safeguard the integrity and effectiveness of recommender systems.

2 Adversarial Attack and Defense on Recommendation System

2.1 Typical Recommendation System

The basic principle: "The basic principle of recommendations is that significant dependencies exist between user and item-centric activity." Charu C.Aggarwal

Two main-types of Recommender System:

- **Content-Based Filtering**: Recommending items to users by analyzing the attributes or features of items and comparing them to a user's profile or historical preferences.
- **Collaborative Filtering**: There are two main approaches
  - **Neighborhood-based Collaborative Filtering**: Exploiting the similarity among users to recommend items. The model allows users to discover

new knowledge (known by similar users).

    * **User-Based collaborative filtering (User-User collaborative filtering)**: Recommending items to a user based on the preferences and behaviors of users with similar profiles.
    * **Item-Based collaborative filtering(Item-Item collaborative filtering)**: Recommending items to a user by finding items that are similar to those the user has interacted with.
  - **Matrix Factorization Collaborative Filtering**: Main idea behind Matrix Factorization for Recommendation Systems:
    * Latent features exist that describe the relationship between items and users.
    * For example: with the movie suggestion system, the latent feature can be criminal, political, action, comedy, ...

- **Hybrid recommender system**: Besides Content-Based Filtering and Collaborative Filtering, in the modern recommender system, Hybrid recommender systems have gained popularity in recent years due to their ability to address the limitations of individual recommendation techniques and provide more accurate and personalized recommendations

2.2 Adversarial Attack

The most common reason to cause a malfunction in a machine learning model; an adversarial attack, might entail presenting a model with inaccurate or misrepresentative data as its training or introducing maliciously designed data to deceive an already trained model.

These attacks can be considered a very acute version of Anomalies in the dataset, directed maliciously from the get to affect a machine learning model. To understand better, most machine learning techniques are primarily designed to work on specific problem sets, assuming that the training and test data are generated from the same statistical distribution. Still, sometimes this assumption can be exploited by some users deliberately to mess up your MLOps pipeline. Users often use these attacks to manipulate your model's performance, affecting your product and reputation.

Adversarial attacks on machine learning require that augmentation and additions be introduced in the model pipeline, especially when the model holds a vital role in situations where the error window is very narrow. For example in Fig. 1, an adversarial

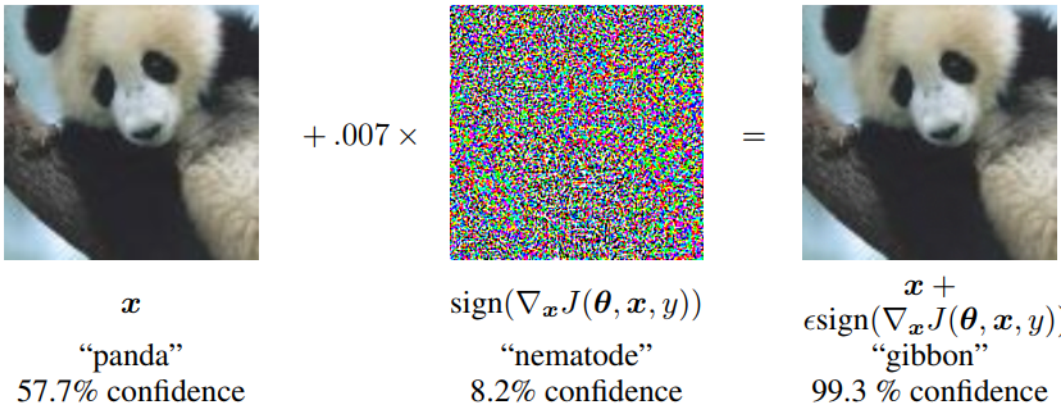


Figure 1: Adding a small vector perturbation can change the prediction of a classification model [1].

attack could involve feeding a model false or misleading data while training or adding maliciously prepared data to trick an already trained model.

To break this down, what the said "perturbation" has done to the panda image is that it has taken into account how the feature extractor in the model will filter the image and effectively change or influence the values of those specific pixels to classify the image wrongly completely.

Some more examples where these attacks can destroy the pipeline can be seen in the Automation Industry, where something like putting wrong stickers on the street can off-put an autonomous vehicle and confuse the decision-making module for horrible outcomes.

We can classify attacks against a ML model along three main dimensions:

- **Attack timing**: which part of the model pipeline is affected
- **Attack goal** : which is the expected output
- **Attacker's knowledge** : what the attacker know about the model: data, algorithm.

2.2.1 Attack Timing

As illustrated in Fig. 2, an adversary can attack a ML model at two main stages of the learning pipeline, during training or production. These two categories of attacks are

respectively known as (i) training-time attack (a.k.a. causative or poisoning attack) [2] and, ii) inference-time attack (a.k.a. exploratory or evasion attack) [3].

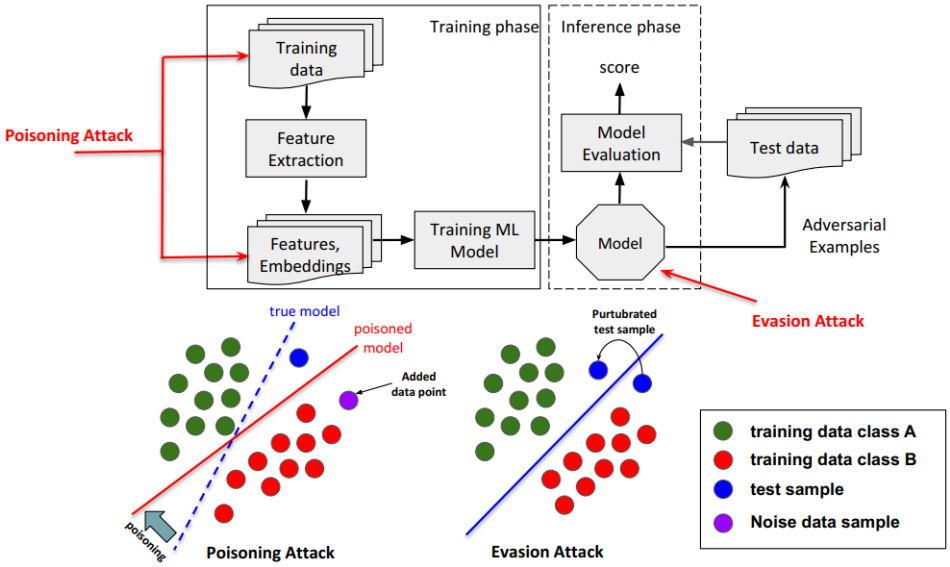


Figure 2: The distinction between evasion attacks and poisoning attacks to a classification model [4].

- **Poisoning attack** Data poisoning attacks are realized by injecting false data points into the training data with the goal to corrupt/degrade the model (e.g., the classifier). Poisoning attacks have been explored in the literature for a variety of tasks [5], such as (i) attacks on binary classification for tasks such as label flipping or against kernelized SVM [6], (ii) attacks on unsupervised learning such as clustering and anomaly detection [7], and (iii) attacks on matrix completion task in RS [8, 9]. As an example, in the pioneering work by Biggio et al. [2], the authors propose a poisoning attack based on properties of the SVM optimal solution that could significantly degrade the classification test accuracy.

- **Evasion attack** Unlike poisoning attacks, evasion attacks do not interfere with training data. They adjust malicious samples during the inference phase. These attacks are also named decision-time attacks referring to their attempt to evade the decision made by the learned model at test time [5]. For instance, evasion attacks can be used to evade spam [10], as well as network intrusion [11] detectors. Recently, evasive attacks are conducted by crafting adversarial examples, which are subtle but non-random human-imperceptible perturbations, added to original data to cause the learned model to produce erroneous output. Szegedy et al. [3]

were the first to discover that some carefully selected perturbations that are barely perceptible to the human eye, when added to an image, could lead a well-trained DNN to missclassify the adversarial image with high confidence.

2.2.2 Attack goal

Attacks are conducted for different goals. We can distinguish between two main classes of attack goals: i) untargeted attack and, ii) targeted attack. To provide the reader with an intuitive insight of the mechanism behind adversarial attacks and defense strategies, we define them formally for a classification task [5].

- **Untargeted attack** [3]: The goal of the attacker in untargeted adversarial attack (misclassification) is to add a minimal amount of perturbation δ on the input sample x such that it can cause incorrect classification [3]. Given $f(x;\theta) = y$, an Untargeted Adversarial Attack is formulated as Eq.1.

$$\begin{aligned} &\min_{\delta} ||\delta|| \\ &\text{s.t,:} f(x;\theta) \neq y, x+\delta \in [0,1]^n \end{aligned} \tag{1}$$

  The second constraint $x + \delta \in [0, 1]^n$ is a value-clipping constraint needed for images, to bound the adversarial samples into to a predefined range so that the images remain visible after adversarial attack. Alternatively, we can formulate the problem as an unconstrained optimization problem where the goal of the attacker is to maximize the loss between the perturbed sample $x + \delta$ and true class y.

$$\max_{\delta:||\delta||\leq\epsilon} l(f(x+\delta;\theta), y) \tag{2}$$

  Obviously since adding an unbounded amount of perturbation on the input will eventually lead to a classification error, the goal of the attacker is to minimize a norm-constrained form of perturbation, that is $||\delta|| \leq \epsilon$ for some exogenously given δ.

  In the context of DNN, the above attacks are categorized based on the norm used to represent the magnitude of the perturbation according to the following norm types [5]: l_0, l_1 and l_2 and l_∞.

- **Targeted Attack** [3]: The goal of the attacker in targeted adversarial attack is to perturb the input by adding a minimum amount of perturbation θ such that it can force the model to miss-classify the perturbed sample into an illegitimate target class (aka mis-classification label).

Given $f(x;\theta) = y$ with $y \neq y_t$, the problem is formulated as Eq. 3

$$\begin{aligned} &\min_{\delta} ||\delta|| \\ &\text{s.t,:} f(x;\theta) = y_t \end{aligned} \tag{3}$$

Similarly, the above problem can be expressed as a unconstrained optimization problem in Eq. 4.

$$\min_{\delta:||\delta||\leq\epsilon} l(f(x+\delta;\theta), y_t) \tag{4}$$

2.2.3 Attacker's knowledge

Black box Attack assumes that the attacker can query the model but does not have access to the model's architecture, parameters, training dataset etc. (ie.:one where we only know the model's inputs, and have an oracle we can query for output labels or confidence scores)

A popular black box attack technical is CopyAttack:

- A reinforcement learning based black-box attack method that harnesses real users from a source domain by copying their profiles into the target domain with the goal of promoting a subset of items.
- CopyAttack is constructed to both efficiently and effectively learn policy gradient networks that first select, and then further refine/craft, user profiles from the source domain to ultimately copy into the target domain. CopyAttack's goal is to maximize the hit ratio of the targeted items in the Top-k recommendation list of the users in the target domain.

CopyAttack is constructed to both efficiently and effectively learn policy gradient networks that first select, and then further refine/craft, user profiles from the source domain to ultimately copy into the target domain. CopyAttack's goal is to maximize the hit ratio of the targeted items in the Top-k recommendation list of the users in the target domain.

White box Attack assumes that the attacker knows everything about the model's architecture, parameters, training dataset etc. The most common attack types so far exploited in the community of RS are fast gradient sign attack (FGSM) [1] and Carlini and Wagner (C&W) attacks, which belong to l_∞- and l_2-norm attack types respectively.

- **Fast Gradient Sign Method (FGSM)** [1] utilizes the sign of the gradient of the loss function to find perturbation that maximizes the training loss (for untargeted case)

$$\delta = \epsilon \ldots \text{sign}(\nabla_x l(f(x;\theta), y)) \tag{5}$$

where ϵ (perturbation level) represents the attack strength and ∇_x is the gradient of the loss function w.r.t. input sample x. The adversarial example is generated as $x^{adc} = x + \delta$. FGSM applies an l_∞- bound constraint $||\delta||_\infty \leq \epsilon$ with the original idea to encourage perceptual similarity between the original and perturbed samples. The unconstrained FGSM aims to find perturbation that would increase/maximize the loss value. The corresponding approach for targeted FSGM [12] is

$$\delta = \epsilon \ldots \text{sign}(\nabla_x l(f(x;\theta), y_t)) \tag{6}$$

where the goal is to maximize the conditional probability $p(y_t|x)$ for a given input x.

Several variants of the FGSM has been proposed in the literature [13, 14]. For instance, the fast gradient value (FGV) method [15], which instead of using the sign of the gradient vector in FGSM, uses the actual value of the gradient vector to modify the adversarial change, or basic iterative method (BIM) [12] (a.k.a iterative FGSM) that applies FGSM attack multiple times iteratively using a small step size and within a total acceptable input perturbation level.

- **Carlini and Wagner (C&W) attack** [16] is one of the most effective attack models. The core idea of C&W attack is to replace the standard loss function — e.g., typically cross-entropy — with an empirically-chosen loss function and use it in an unconstrained optimization formulation given by

$$\min_\delta ||\delta||_p^p + c\dot{l}(x + \delta, y_t) \tag{7}$$

where $\dot{l}()$ is the candidate loss function. The C&W attack has been used with several norm-type constraints on perturbation l_0, l_2, l_∞ among which the l_2-bound constraint has been reported to be most effective [17].

2.3 Defense against Adversarial Attack

The method of defense against adversarial attack can be classified as two main approaches: proactive and reactive [4]. While reactive countermeasures try to detect attack [**?**], reconstruct or verify the output, proactive methods increase the robustness of the learning model. This approach gets the attention of many researchers. The

prominent methods used are the robust optimization [1, 12] and the distillation method [18, 19, 20].

At the heart of the robust optimization method is the assumption that every sample in the training data D can be a source for adversarial behavior. It performs an ERM against a specific adversary on each sample in D and applies a zero sum-game s leading to the following robust optimization framework.

2.3.1 Robust Optimization

Robust Optimization enhances the model with the assumption that every sample in the training data D can be a source for adversarial behavior. It design a minimax game between the prediction and attack adversaries. With the optimal attack model, recommendation model try to give the true output. The general loss can be formulated in 8

$$L = \min_{\Theta} \sum_{(x_i, y_i) \in D} \max_{\delta : ||\delta|| \leq \epsilon} l(f(x_i + \delta; \Theta), y_i) \tag{8}$$

[1] proposed adversarial training is to build a robust model from ground-up on a training set augmented with adversarial examples. Adversarial regularization is one of the mostly investigated techniques for adversarial training, which utilizes an approximation of the worst-case loss function as the regularizer in 9.

$$L_{AT} = \underbrace{\min_{\Theta} \sum_{(x_i, y_i) \in D} [l(f(x_i; \Theta), y_i) + \underbrace{\max_{\delta : ||\delta|| \leq \epsilon} l(f(x_i + \delta; \Theta), y_i)}_{\textit{optimal attack model}}]}_{\textit{optimal robustness-preserving prediction}} \tag{9}$$

2.3.2 Defensive Distillation

Defensive Distillation train distilled model using the label predicted by the initial model. It explicit relative information about classes prevents models from fitting too tightly to the data, and contributes to a better generalization around training points.

For example in Fig**??**, distillation training process for classification task included in two steps:

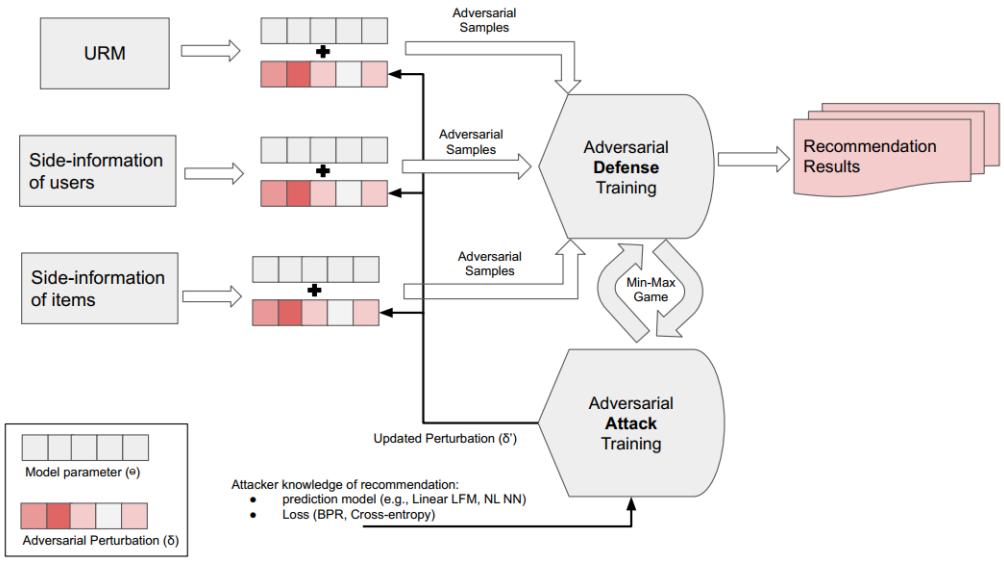


Figure 3: A notional view of Adversarial Recommendation Framework integrating the adversarial perturbations on users and items, and their side information, model parameters. [4]

- Train an initial (or teacher) network F on data X with a softmax temperature of T.
- Use the probability vector $F(X)$, which includes additional knowledge about classes compared to a class label, predicted by network F to train a distilled network F^d at temperature T on the same data X.

3 Adversarial Personalized Ranking

First the matrix factorization model for recommendation is described. Next the pairwise learning method Bayesian Personalized Ranking is shortly recapitulated. The novel contribution of this section is to empirically demonstrate that the MF model optimized by BPR (a.k.a. MF-BPR) is not robust and is vulnerable to adversarial perturbations on its parameters.

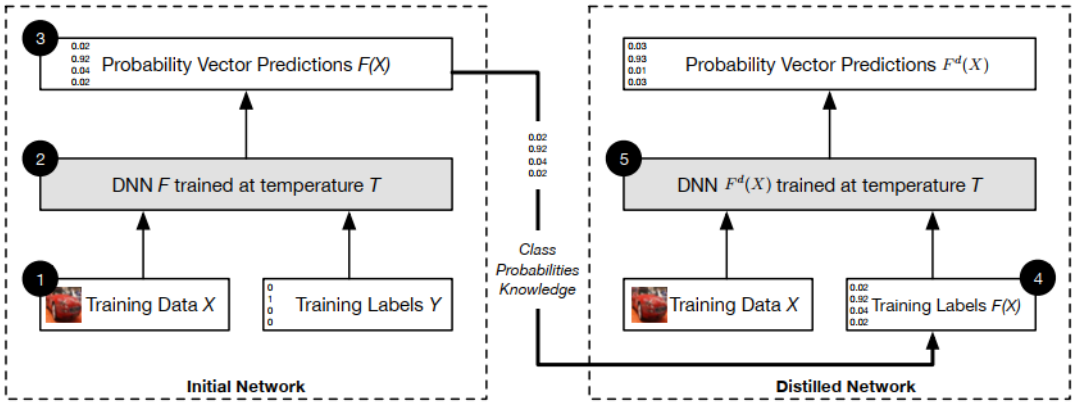


Figure 4: An overview of defense mechanism based on distillation knowledge using probability vectors [20].

3.1 Matrix Factorization

MF has been recognized as the basic yet most effective model in recommendation since several years [21, 22, 23]. Being a germ of representation learning, MF represents each user and item as an embedding vector. The core idea of MF is to estimate a user's preference on an item as the inner product between their embedding vectors.

Formally, let u denote a user and i denote an item, then the predictive model of MF is formulated as: $\hat{y}_{ui}(\Theta) = p_u^T q_i$, where $p_u \in R^K$ and $q_i \in R^K$ denote the embedding vector for user u and item i, respectively, and K is the size of embedding vector also called as embedding size. Θ denotes the model parameters of MF, which is consisted of all user embedding and item embedding vectors, i.e., $\Theta = \{p_u\}_{u \in U}, \{q_i\}_{i \in I}$, where U and I denote the set of all users and items, respectively. We use P and Q to denote the embedding matrix $P = \{p_u\}_{u \in U}, Q = \{q_i\}_{i \in I}$ for short.

3.2 Bayesian Personalized Ranking

BPR is a pairwise L2R method and has been widely used to optimize recommender models towards personalized ranking [28]. Targeting at learning from implicit feedback, it assumes that observed interactions should be ranked higher than the unobserved ones. To this end, BPR maximizes the margin between an observed interaction and its unobserved counterparts. This is fundamentally different from pointwise methods [21, 24] that optimize each model prediction towards a predefined groundtruth. Formally,

the objective function (to be minimized) of BPR is:

$$L_{BPR}(D|\Theta) = \sum_{(u,i,j)\in D} -\ln \sigma(\hat{y}_u i(\Theta) - \hat{y}_u j(\Theta)) + \lambda_\Theta ||\Theta||^2 \tag{10}$$

where $\sigma()$ is the sigmoid function, λ_Θ are model specific regularization parameters to prevent overfitting, and D denotes the set of pairwise training instances $D := (u, i, j)|i \in I_u^+, u \in I_j^-$, where I_u^+ denotes the set of items that u has interacted with before, and I denotes the whole item set. Since the number of training instances in BPR is very huge, the optimization of BPR is usually done by performing stochastic gradient descent (SGD). After obtaining parameters, we can get the personalized ranked list for a user u based on the value of $\hat{y}_u i(\Theta)$ over all items.

Owing to its rationality and ease of optimization, BPR has been used in a wide variety of scenarios, such as multimeida [25], clothing [26], heterogeneous information sources [27], and plays a important role in optimizing recommender models. It is worth noting that the behavior of BPR can be interpreted as a classifier — given a triplet (u, i, j), it determines whether (u, i) should have a higher score than (u, j). Under this interpretation, a positive instance of (u, i, j) means that $\hat{y}_u i$ should be larger than $\hat{y}_u j$ as much as possible to get the correct label of $+1$; and vice versa, a negative instance can be seen as having a label of 0.

3.3 Adversarial Personalized Ranking

[28] designed a new objective function such that by optimizing it, the recommender model is both suitable for personalized ranking and robust to adversarial perturbations. Due to the rationality of the BPR pairwise objective in personalized ranking, the authors choosed it as the building block. To enhance the robustness, the model have to perform well even when the adversarial perturbations are presented, in 11. To achieve this, a optimization is added into the model to minimize the BPR objective function with the perturbed parameters.

$$\begin{aligned} L_{APR}(D|\Theta) &= L_{BPR}(D|\Theta) + \lambda L_{BPR}(D|\Theta + \Delta_{adv}) \\ \text{where } \Delta_{adv} &= arg \max_{\Delta, ||\Delta|| \leq \epsilon} \lambda L_{BPR}(D|\Theta + \Delta_{adv}) \end{aligned} \tag{11}$$

where Δ denotes the perturbations on model parameters, $\epsilon \geq 0$ controls the magnitude of the perturbations, and $\hat{\Theta}$ denotes the current model parameters. In this formulation, the adversarial term $\lambda L_{BPR}(D|\Theta + \Delta_{adv})$ can be seen as regularizing the model by

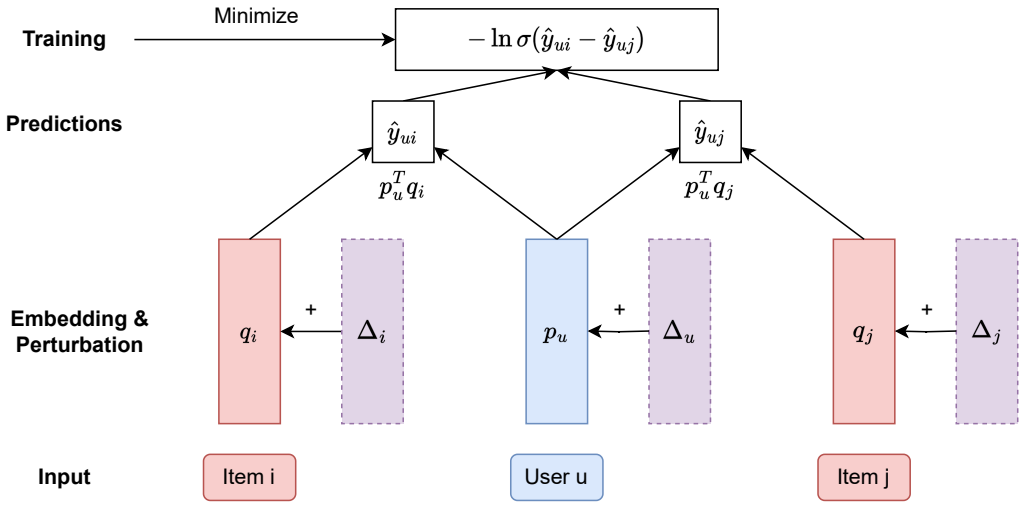


Figure 5: Adversarial Training to Bayesian Personalized Ranking

stabilizing the classification function in BPR. As such, we also call it as adversarial regularizer and use λ to control its strength. As the intermediate variable Δ maximizes the objective function to be minimized by Θ, the training process of APR can be expressed as playing a minimax game:

$$\Theta^*, \Delta^* = arg \min_{\Theta} \max_{\Delta, ||\Delta|| \leq \epsilon} L_{BPR}(D|\Theta) + \lambda L_{BPR}(D|\Theta + \Delta) \tag{12}$$

where the learning algorithm for model parameters Θ is the minimizing player, and the procedure for getting perturbations Δ acts as the maximizing player, which aims to identify the worst-case perturbations against the current model. The two players alternately play the game until convergence. Since the focus of APR is to get a good recommender model, in practice we can determine when to stop the adversarial training by tracking how does the model perform on a validation set.

We can see that, similar to BPR, our formulation of APR leads to a general learning framework which is model independent. As long as the underlying model $\hat{y}_{ui}(\Theta)$ is differentiable, it can be learned under our APR framework using backpropagation and gradientbased optimization algorithms. There are two hyper-parameters — ϵ and λ— to be specified in APR in addition to the ones in BPR. In what follows, we propose a generic solution for APR based on SGD

4 Distillation for Matrix Factorization

We propose a distillation method for MF-BPR. Firstly, we train a MF-BPR as initial or teacher model. For each pair (u, i) with every user and item in datset (not all are positive user-item pair), initial model generate a rating r. The predicted rating r is regard as ground truth value and the distillation model have to estimate. Now, we transform the problem to a regression task and using Mean Square Error as objective function in training process. Moreover, we propose a adversarial training version when apply adversarial term for loss function. They are illustrated in figure 6.

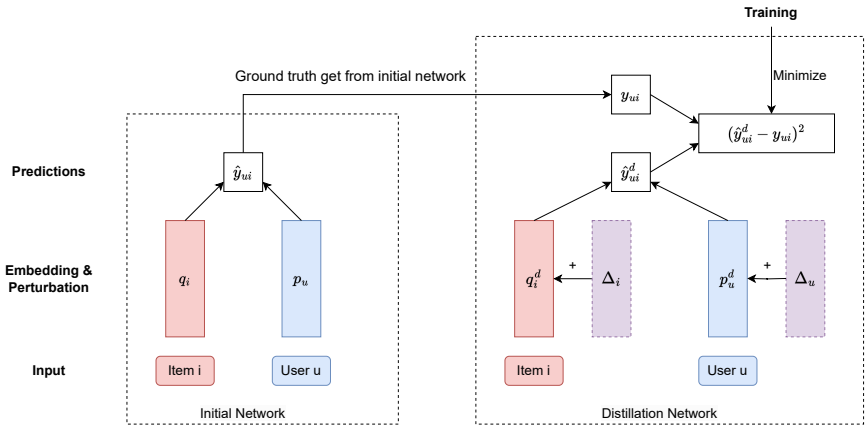


Figure 6: Distillation approach

5 Experiments and Results

5.1 Dataset

For experiments, we using three datasets: Movie Lens and Pinterest. The general information about these datasets are described in table 1.

- **Yelp**: This is the Yelp Challenge data of user ratings on businesses. We use the filtered subset created by [29] for evaluating item recommendation. We find that a user may rate an item multiple times at different timestamps. Since a recommender system typically aims to recommend items that a user did not consume before, we further merge repetitive ratings to the earliest one. This can also avoid a testing interaction appearing in the training set.

- **Book-Crossing**: Book Crossing dataset were collected by Cai-Nicolas Ziegler [30] in a 4-week crawl (August / September 2004) from the Book-Crossing community with kind permission from Ron Hornbaker, CTO of Humankind Systems. This data has been organized and cleaned up by removing all users and items who had less than 20 and 10 interactions, receptively, items that have no information and separated in files the explicit and implicit interactions.

Table 1: Description of three dataset.

Dataset	**No. Interaction**	**No. Item**	**No. User**	**Sparsity**
Yelp	730,790	25,815	25,677	99.89%
Book-crossing	62,657	14,684	1,295	96.34%

5.2 Scenarios

To evaluate the effciency of defense method, we performed different experiments scenarios:

1. Impact of adding random and adversarial perturbation to Recommendation System: MF-BPR, APR and distialltion model.
2. Performance comparison between MF-BPR and APR.

We employed three criteria: Hit Ratio (HR), Normalized Discounted Cumulative Gain (NDCG) and Area under the ROC Curve (AUC).

- Hit Ratio (HR): is a recall-based metric, measuring whether the testing item is in the top-K list.
- Normalized Discounted Cumulative Gain (NDCG): is position-sensitive, which assigns higher score to hits at higher positions.
- Area under the ROC Curve (AUC): the probability that the model ranks a random positive example more highly than a random negative example

For both metrics, larger values indicate better performance. We report the average score for all users, and perform one-sample paired t-test to judge the statistical significance where necessary.

Table 2: Hyper-parameter setting

parameter	value
lambda	1
epoch	1000
embedding size	64
batch size	512
learning rate	0.05

5.3 Impact of Perturbation

In this experiments, we implemented recommendation models: MF-BPR, APR and distillation version on small dataset Book-Crossing. Beside that, we trained separately noise model, plays a role as attacker. There are two types of noise: random noise and adversarial noise. We set up many values of noise magnitude ϵ to evaluate the impact of perturbation attack to the recommendation model.

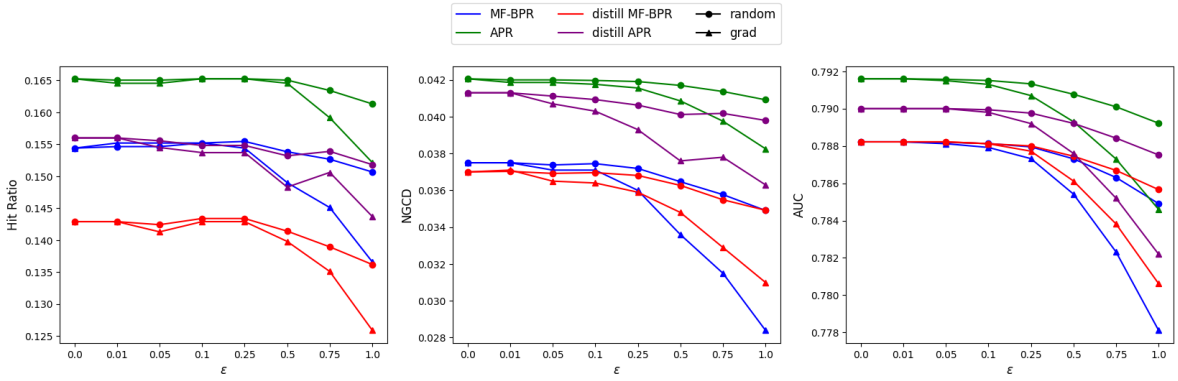


Figure 7: Performance of the models on Book-crossing dataset decreased by adding perturbation.

The Fig. 7 indicate that more perturbation magnitude, the worse models perform. In regular scenarios, the results of MF-BPR is equivalent the models with adversarial training. When the perturbation magnitude increase, the performance of all decreased but MF-BPR is worse than the others. Under an attack MF-BPR is more vulnerable than the models using defense method.

On the other hand, we found that adversarial perturbation is more harmful than the random. It indicate that, a white box attack always is dangerous than a black box.

About the distillation method, we found that the small model size make the dis-

tillation just tried to mimic the initial model. Therefore, their improvement is not significant. We think it need more adjustment to be better and it may be the reason why distillation approaches are not popular as robust optimization.

5.4 Performance Comparison

For more details, we compare the performance of MF-BPR and APR with two version of perturbation (random and gradient). The table 3 and 4 indicate the results on dataset Book-crossing and Yelp, respectively.

Table 3: Top-K recommendation performance at K = 50 and K = 100 in Book-crossing Dataset

	HR		NDGC		AUC
	K=50	K=100	K=50	K=100	
MF-BPR	0.0988	0.1544	0.0310	0.0375	0.7882
AMF-random	0.1066	0.1529	**0.0358**	0.0421	0.7849
AMF-grad	**0.1112**	**0.1653**	0.0336	**0.0424**	**0.7916**

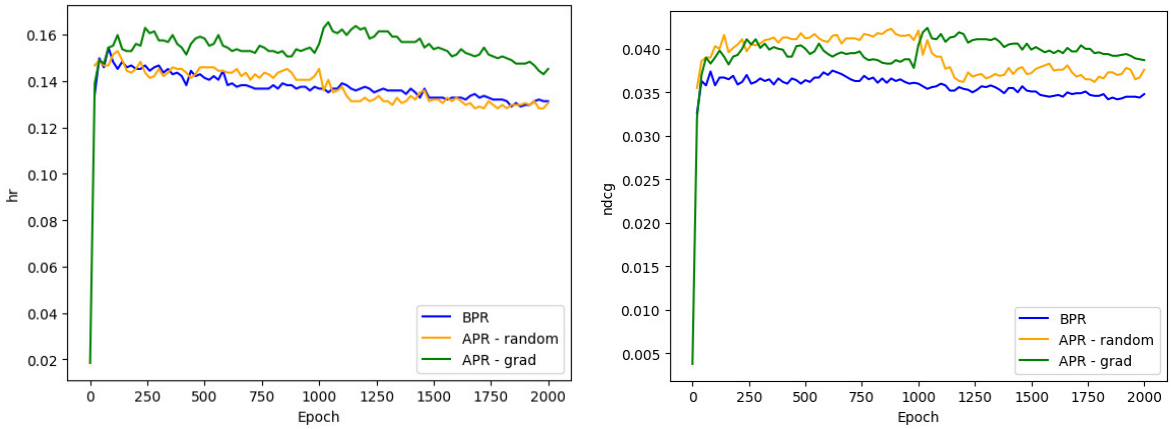
Table 4: Top-K recommendation performance at K = 50 and K = 100 in Yelp Dataset

	HR		NDGC		AUC
	K=50	K=100	K=50	K=100	
MF-BPR	0.1227	0.1906	0.0373	0.0483	**0.9268**
AMF-random	0.1227	0.1876	**0.0383**	0.0488	0.9173
AMF-grad	**0.1272**	**0.2027**	**0.0383**	**0.0504**	0.9190

As what we found in 5.3, the random perturbation is not harmful as adversarial. Therefore, adding random perturbation attack in training process not give significant changes, it's just like a normal regularization. Obviously, adversarial perturbation improve the results in almost criteria, included in important two are HR and NDGC.

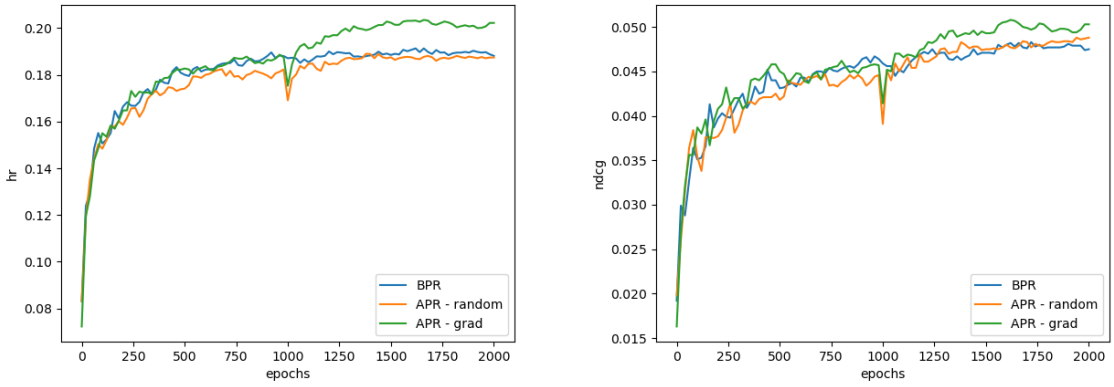
On the other hand, we visual the top-100 HR and NDGC in training process of each model. The figure 8 and 9 present the metrics at each 20 epochs.

In small dataset Book-crossing, all model quickly achive optimal and the after epoch seem to be not better. While in large dataset Yelp, they grow epoch by epoch and reach the optimal at last.



(a) Hit rate (b) NDCG

Figure 8: Performance comparision of MF-BPR and APR on Book-Crossing dataset



(a) Hit rate (b) NDCG

Figure 9: Performance comparision of MF-BPR and APR on Yelp dataset

6 Conclustion

In this project, we study about recommendation system, attack and defense method . Especially, we focus on Matrix Factorization- Bayesian Personalized Ranking model, the way Fast Gradient Sign Method attack it and how to defense against by Adversarial Training and Distillation. Moreover, we made effort to implement them, so that we can verify the result of the researches. Our experiments explore that gradient perturbation is more dangerous random, which means if the attackers know more about the system, they can make it more depress. Beside that, in some case, the distillation is not efficient so it not get muach attention of researchers. Robust optimization is popular method, achieve high improvement under normal and attack scenarios.

References

[1] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[2] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," *ICML*, 2012.

[3] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *ICLR*, 12 2013.

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

[5] Y. Vorobeychik, M. Kantarcioglu, R. Brachman, P. Stone, and F. Rossi, *Adversarial machine learning*. Springer, 2018, vol. 12.

[6] H. Xiao, H. Xiao, and C. Eckert, "Adversarial label flips attack on support vector machines," in *ECAI 2012*. IOS Press, 2012, pp. 870–875.

[7] B. Biggio, K. Rieck, D. Ariu, C. Wressnegger, I. Corona, G. Giacinto, and F. Roli, "Poisoning behavioral malware clustering," in *Proceedings of the 2014 workshop on artificial intelligent and security workshop*, 2014, pp. 27–36.

[8] Y. Deldjoo, T. Di Noia, and F. A. Merra, "Assessing the impact of a user-item collaborative attack on class of users," *arXiv preprint arXiv:1908.07968*, 2019.

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

[10] Z. Jorgensen, Y. Zhou, and M. Inge, "A multiple instance learning strategy for combating good word attacks on spam filters." *Journal of Machine Learning Research*, vol. 9, no. 6, 2008.

[11] K. Wang, J. J. Parekh, and S. J. Stolfo, "Anagram: A content anomaly detector resistant to mimicry attack," in *International workshop on recent advances in intrusion detection*. Springer, 2006, pp. 226–248.

[12] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," *arXiv preprint arXiv:1611.01236*, 2016.

[13] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, "Adversarial attacks and defences: A survey," *arXiv preprint arXiv:1810.00069*, 2018.

[14] R. R. Wiyatno, A. Xu, O. Dia, and A. De Berker, "Adversarial examples in modern machine learning: A review," *arXiv preprint arXiv:1911.05268*, 2019.

[15] A. Rozsa, E. M. Rudd, and T. E. Boult, "Adversarial diversity and hard positive generation," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2016, pp. 25–32.

[16] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 ieee symposium on security and privacy (sp)*. Ieee, 2017, pp. 39–57.

[17] ——, "Defensive distillation is not robust to adversarial examples," *arXiv preprint arXiv:1607.04311*, 2016.

[18] X. Chen, Y. Zhang, H. Xu, Z. Qin, and H. Zha, "Adversarial distillation for efficient recommendation with external knowledge," *ACM Transactions on Information Systems (TOIS)*, vol. 37, no. 1, pp. 1–28, 2018.

[19] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[20] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE symposium on security and privacy (SP)*. IEEE, 2016, pp. 582–597.

[21] I. Bayer, X. He, B. Kanagal, and S. Rendle, "A generic coordinate descent framework for learning from implicit feedback," in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 1341–1350.

[22] H. Zhang, F. Shen, W. Liu, X. He, H. Luan, and T.-S. Chua, "Discrete collaborative filtering," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2016, pp. 325–334.

[23] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, pp. 426–434.

[24] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 173–182.

[25] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, "Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention," in *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2017, pp. 335–344.

[26] W. Yu, H. Zhang, X. He, X. Chen, L. Xiong, and Z. Qin, "Aesthetic-based clothing recommendation," in *Proceedings of the 2018 world wide web conference*, 2018, pp. 649–658.

[27] Y. Zhang, Q. Ai, X. Chen, and W. B. Croft, "Joint representation learning for top-n recommendation with heterogeneous information sources," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 1449–1458.

[28] X. He, Z. He, X. Du, and T.-S. Chua, "Adversarial personalized ranking for recommendation," in *The 41st International ACM SIGIR conference on research & development in information retrieval*, 2018, pp. 355–364.

[29] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2016, pp. 549–558.

[30] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen, "Improving recommendation lists through topic diversification," in *Proceedings of the 14th international conference on World Wide Web*, 2005, pp. 22–32.