Certifiable Robustness to Discrete Adversarial Perturbations for Factorization Machines

Yang Liu
Sun Yat-sen university
liuy296@mail2.sysu.edu.cn

Xianzhuo Xia
Sun Yat-sen university
xiaxzh@mail2.sysu.edu.cn

Liang Chen*
Sun Yat-sen university
chenliang6@mail.sysu.edu.cn

Xiangnan He
University of Science and Technology of China
xiangnanhe@gmail.com

Carl Yang
Emory University
j.carlyang@emory.edu

Zibin Zheng
Sun Yat-sen university
zhzibin@mail.sysu.edu.cn

ABSTRACT

Factorization machines (FMs) have been widely adopted to model the discrete feature interactions in recommender systems. Despite their great success, currently there is no study of their robustness to discrete adversarial perturbations. Whether modifying a certain number of the discrete input features has a dramatic effect on the FM's prediction? Although there exist robust training methods for FMs, they neglect the discrete property of input features and lack of an effective mechanism to verify the model robustness.

In our work, we propose the first method for the certifiable robustness of factorization machines with respect to the discrete perturbation on input features. If an instance is certifiably robust, it is guaranteed to be robust (under the considered space) no matter what the perturbations and attack models are. Likewise, we provide non-robust certificates via the existence of discrete adversarial perturbations that change the FM's prediction. Through such robustness certificates, we show that FMs and the current robust training methods are vulnerable to discrete adversarial perturbations. The vulnerability makes the outcome unreliable and restricts the application of FMs. To enhance the FM's robustness against such perturbations, a robust training procedure is presented whose core idea is to increase the number of instances that are certifiably robust. Extensive experiments on three real-world datasets demonstrate that our method significantly enhances the robustness of the factorization machines with little impact on predictive accuracy.

CCS CONCEPTS

• **Security and privacy**; • **Information systems** → *Recommender systems*;

KEYWORDS

Robustness; Adversarial Examples; Factorization Machine; Sparse Prediction

ACM Reference Format:
Yang Liu, Xianzhuo Xia, Liang Chen, Xiangnan He, Carl Yang, and Zibin Zheng. 2020. Certifiable Robustness to Discrete Adversarial Perturbations for Factorization Machines. In *43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), July 25–30, 2020, Virtual Event, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397271.3401087

1 INTRODUCTION

Due to the strong ability of handling discrete features, factorization machines (FMs) [26, 27] have been widely employed in many high impact applications such as recommender systems [7, 33] and computational advertising [15]. The input features for these applications are mostly discrete and categorical. To deal with such types of features, a common solution is to convert them to binary features via one-hot encoding [17] (e.g. the gender of users) or multi-hot encoding (e.g. the historical items of users). Since the number of possible values is large, the resulting discrete feature vector can be high-dimensional and sparse. To build an effective model from such sparse data, FMs incorporate the interactions between features. Specifically, FMs generate new features by combining multiple individual features. For examples, a new feature $clothes\_color = \{coat\_pink, sweater\_pink, coat\_red, sweater\_red\}$ is obtained by combining $clothes = \{coat, sweater\}$ and $color = \{pink, red\}$.

Recent studies demonstrate that machine learning models, including graph neural networks [10, 37], convolutional neural networks [13], and decision trees [5], are vulnerable to slight and deliberate perturbations (known as adversarial perturbations). By slightly modifying the input data (e.g. an image), two very similar instances are classified into completely different classes. Such unreliable results significantly hinder the applicability of these models. So far, the questions of adversarial perturbations on the factorization machine has not been addressed: Can factorization machines be easily fooled? How reliable are their results? Considering the perturbations on the instance's features which do not reflect users' preference are easy to be injected by attackers (e.g. fraudsters manipulate online reviews [20, 37]), the questions are highly important and necessary to be solved.

The existing robust factorization machine [24] considers the environmental noise in user signals. They model such noise by

*Liang Chen is the corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SIGIR '20, July 25–30, 2020, Virtual Event, China
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-8016-4/20/07…$15.00
https://doi.org/10.1145/3397271.3401087

associating an uncertainty vector (e.g. Gaussian distribution) on the input features. To enhance the model robustness under the noise, they seek a solution that remains feasible for all possible perturbations on input features via minimizing the worst-case loss. However, there are two limitations of current robust factorization machine:

- **Neglecting inputs' discrete property.** The perturbations they considered are continuous intervals (e.g. [0, 0.05]). Such an assumption is not applicable to the FM since most features in the FM's application is discrete/binary. The possible perturbations under the binary setting are {-1, 1}. Therefore, the worst-case loss they minimize is insufficient to model the real worst-case perturbation which leads to a sub-optimal solution.
- **Lacking the robustness certificate.** At present, there are no effective mechanisms that can verify whether a given FM is robust. Existing work [24] compares the model performance under specific noises. Given such a large perturbation space, this approach is not enough to verify the robustness.

To verify the robustness of the FM, the core idea is to generate the worst-case discrete perturbations given a certain perturbation space (i.e. the number of changed features in our binary setting). If even under the worst-case the FM is robust, the FM is provably robust. Otherwise, we find the discrete adversarial perturbation which can change the prediction of the FM. Nevertheless, the difficulty is the high time complexity. We find that the computation of the worst-case discrete perturbation needs to enumerate all possible perturbations due to the dependencies between features. However, since input features are sparse and high-dimensional which leads to a large perturbation space, the time complexity of such enumeration is unbearable. Furthermore, given the discrete perturbations, we observe that the FM [26] is vulnerable to such perturbations. How to make the FM less sensitive to the discrete perturbations is another challenge.

To tackle the above challenge, we propose the first method for *certifiable robustness* of FMs which approximates the worst-case perturbation. Specifically, we provide: (1) **Robustness certificates**. Given a trained FM and a certain space of perturbation, we give robustness certificates (i.e. robust or non-robust) for input instances. If the instance is certifiably robust, it is guaranteed that no perturbations in the considered space can change the prediction of the instance. We derive the bound of the FM's prediction that can be reached by the worst-case perturbation. If the bound does not change the predicted label of the instance, then the FM is provably robust. Similarly, an instance is certifiably non-robust if there exist discrete adversarial perturbations in the considered perturbation space that can change the instance's prediction. We approximate the existence of such perturbations by sequentially selecting the most promising feature towards the worst-case until the perturbation budget is exhausted or the prediction is changed. (2) **Robust training**. To enhance the robustness under the discrete adversarial perturbation, we propose a training approach which maximizes the number of instances that are certifiably robust.

Overall, our contributions are:

- We study the robustness of factorization machines and show that they are vulnerable to the discrete adversarial perturbations.
- By approximating the worst-case perturbation, we propose a novel provable framework to analyze the robustness of factorization machines.
- We propose robust training based on our robustness certificates which maximize the number of the certifiably robust instances.
- We empirically show that our certificates are tight and extensive experiments conducted on three real-world datasets demonstrate that our robust training can improve the robustness of FMs while has little impact on classification performance.

2 PRELIMINARIES

First the model formulation and feature representation are introduced. Then the problem and the challenge are elaborated. We use bold uppercase letters to denote matrices (e.g., $\mathbf{W}$), bold lowercase letters to denote vectors (e.g., $\mathbf{w}$), and non-bold letters to denote scalars or indices (e.g., w). The uppercase calligraphic symbols (e.g., $\mathcal{W}$) stand for sets.

2.1 Factorization Machine

Factorization machine [26] is a popular learning paradigm for sparse data which enhances linear regression with feature interactions. An example of sparse features that consists of three fields is displayed as follows.

$$\underbrace{[1, 1, ..., 0]}_{\text{Historical Items}} \quad \underbrace{[0, 1, ..., 0]}_{\text{User ID}} \quad \underbrace{[1, 0, ..., 0]}_{\text{Item Brand}}, \tag{1}$$

where the user id and item brand are one-hot encoding and the historical items are multi-hot encoding.

In our work, we consider the second-order interaction between features. To learn from the sparse data, FMs project each feature to a latent space and use the inner product of corresponding feature embeddings to compute their interaction weight. Following the previous work [24], we add self-interaction terms to the model. Formally, the FM model is defined as follows:

$$f_\theta(\boldsymbol{x}) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d}\sum_{j=i}^{d} < \boldsymbol{v}_i, \boldsymbol{v}_j > x_i x_j, \tag{2}$$

where x_i denotes the i-th component of feature vector $\boldsymbol{x} \in \{0, 1\}^{1\times d}$ and d is the dimension of the feature vector. w_0 represents the global bias and w_i is the weight of i-th feature. $\boldsymbol{v}_i \in \mathbb{R}^{1\times k}$ is the embedding vector of i-th feature and $< \boldsymbol{v}_i, \boldsymbol{v}_j >$ is the inner product which models the interactions between the i-th and j-th feature. θ is used to denote all parameters. According to the previous work [26], the feature interaction term can be reformulated as:

$$\sum_{i=1}^{d}\sum_{j=i}^{d} < \boldsymbol{v}_i, \boldsymbol{v}_j > x_i x_j = \frac{1}{2}\sum_{f=1}^{k}\left[\left(\sum_{j=1}^{d} v_{j,f} x_j\right)^2 + \sum_{j=1}^{d} v_{j,f}^2 x_j^2\right], \tag{3}$$

where $v_{j,f}$ denotes the f-th component of $\boldsymbol{v}_j$.

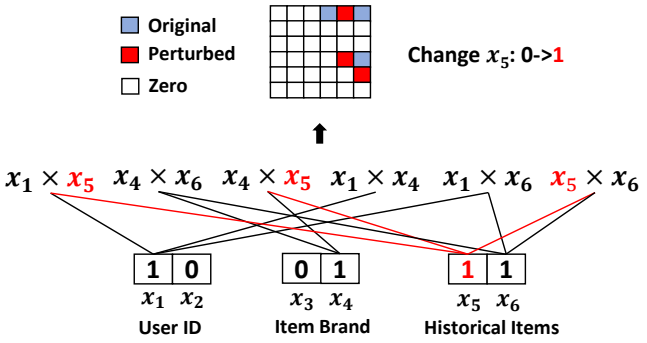

Figure 1: A simple illustration of the adversarial attack. The first-order weight of FM is omitted. The blue and red denotes the original and perturbed feature interactions respectively. The matrices represent all feature interactions. Note that it is not necessary to compute weights of all feature interactions, we list them for better illustration.

2.2 Problem Formulation

In this paper, we model the recommendation problem as binary classification (the label is {1, -1}) whose input data is binary as shown in Section 2.1. Distinct from the previous work [24], the perturbation we considered is to change the 0 components of input features to 1 which is closer to the reality. Since the fields of input features usually contain specific semantics, directly modifying the fields may change the semantics of the instance. For example, by changing the user ID of an instance, its semantic is completely changed and the predicted result may be different. In our work, we focus on perturbing users' historical items. That is, the perturbed user-item interactions are injected into the input features. It is natural to assume that a robust model should not change its predictions even there exit small perturbations on user-item interactions. Figure 1 displays a simple example of the adversarial attack which injects a historical item. The red weights denote the injected item's influence on the FM's second-order weights.

Our first goal is to derive an efficient robustness analyzer for factorization machines. That is, given a trained FM f_θ, a specific instance $\boldsymbol{x}$ (called targeted instance) and a perturbation space, our goal is to provide a robustness certificate which states whether the FM's prediction for the instance is robust or non-robust. If the prediction is certifiably robust, it is guaranteed that no perturbation in the given perturbation space can change its label. In other words, the predicted label will not change even if the input data is perturbed (under the given perturbation space). In contrast, If the prediction is certifiably non-robust, it is guaranteed that there exist adversarial perturbations in considered perturbation space which can change the prediction made for the instance $\boldsymbol{x}$.

Let $\hat{\boldsymbol{x}}$ denotes the perturbed instances and $\mathcal{P}_q(\boldsymbol{x})$ be the set of all possible binary perturbations (i.e. the perturbation space) of the instance $\boldsymbol{x}$ where $q \in \mathbb{N}$ is the perturbation budget. If all $\hat{\boldsymbol{x}} \in \mathcal{P}_q(\boldsymbol{x})$ are classified to the same class of $\boldsymbol{x}$, then the FM is certifiably robust w.r.t the instance $\boldsymbol{x}$. Note that there is no need to define attack models that generate perturbations given a perturbation space. Once an instance is certifiably robust, its label will not change no matter what attack models are (in the given perturbation space). In contrast, if there exists $\hat{\boldsymbol{x}} \in \mathcal{P}_q(\boldsymbol{x})$ and its label is different with the predicted label, then the FM is certifiably non-robust w.r.t the instance $\boldsymbol{x}$. To summarize, the problem is defined as:

Problem 1. *Given a trained FM f_θ, a target instance $\boldsymbol{x}$, and a perturbation space $\mathcal{P}_q(\boldsymbol{x})$, check whether all $\hat{\boldsymbol{x}} \in \mathcal{P}_q(\boldsymbol{x})$ belong to the same class of $\boldsymbol{x}$.*

Note that if we can find the optimal solution of the above problem, an instance that can not be certifiably robust is non-robust. However, in Section 2.3, we show that finding such optimal solution is a NP-complete problem.

2.3 Computational Challenge

To check whether a perturbed instance $\hat{\boldsymbol{x}} \in \mathcal{P}_q(\boldsymbol{x})$ is still classified to the same class, the margin δ defined as follows between its and the true predicted result is required to be computed.

$$\delta = f_\theta(\hat{\boldsymbol{x}}) - f_\theta(\boldsymbol{x}) \tag{4}$$

If the margin δ does not change the instance's sign, it is certifiably robust. If $f_\theta(\boldsymbol{x}) > 0$, the predicted label for the instance $\boldsymbol{x}$ is 1, otherwise is -1. Thus we check whether $f_\theta(\boldsymbol{x}) + \delta$ is in the same interval with $f_\theta(\boldsymbol{x})$. For example, suppose the predicted label of $\boldsymbol{x}$ is 1 and $f_\theta(\boldsymbol{x}) + \delta$ is greater then 0 as well, then such a perturbation δ will not change the label of the instance $\boldsymbol{x}$.

Although it is not difficult to check a perturbed instance, it is non-trivial to check all perturbed instances $\hat{\boldsymbol{x}} \in \mathcal{P}_q(\boldsymbol{x})$. Generally speaking, if even in the worst-case, the predicted label is not changed, the instance is provably robust. In other words, when the predicted label of $\boldsymbol{x}$ is 1, if the sign of $f_\theta(\boldsymbol{x}) + \delta_{min}$ is still 1 where δ_{min} is the minimal δ reached by all possible perturbations, the instance is certifiably robust. Similarly, when the predicted label of $\boldsymbol{x}$ is -1, if the sign of $f_\theta(\boldsymbol{x}) + \delta_{max}$ is still -1 where δ_{max} is the maximal δ of all possible perturbations, the instance is certifiably robust.

However, we show that the worst-case perturbation can not be computed directly. Let $\boldsymbol{x}' \in \{0,1\}^{1\times d}$ be the perturbation vector. That is, $\hat{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{x}'$. Note that not all d components of x are perturbable. Since we consider the perturbation on the user's historical items, only n ($n < d$) components of x are perturbable where n is the number of items. The margin δ can be computed as equation (5). As can be seen from the above formulation, δ can be divided into three parts: the first-order weights of the perturbation vector $\sum_{j=1}^{d} w_j x'_j$, feature interactions between the perturbation vector $\boldsymbol{x}'$ and input features $\boldsymbol{x}$, and feature interactions of the perturbation vector $\boldsymbol{x}'$ itself. Given δ, we want to find the worst-case perturbation $\boldsymbol{x}'$. Since $\boldsymbol{x}'$ is binary, the problem is equivalent to finding a subset of its components $\{\boldsymbol{x}'_j\}_{j=1}^{n}$ (i.e. changing these components to 1 and the remaining components are zero) that lead to the worst-case of δ. To find an optimal solution, the challenge is the computation of $\left(\sum_{j=1}^{n} v_{j,f} x'_j\right)^2$. Note that this is the same as finding a subset of $\{v_{j,f}\}_{j=1}^{n}$ which sum to the maximal or minimal of δ and the subset sum problem is a well-known NP-Complete problem. However, enumerating all possible subsets is necessary to find the optimal solution and such solution is not practical since the input feature

of the FM is usually sparse and high-dimensional which results in high time complexity.

As such, our major challenge is to derive an approximation method to reduce the time complexity while certificates the robustness of instances as much as possible given the trained model parameters and perturbation space.

$$
\begin{aligned}
\delta &= \sum_{j=1}^{d} w_j x'_j + \frac{1}{2}\sum_{f=1}^{k}\sum_{j=1}^{d} v_{j,f}^2 x_j'^2 \\
&+ \frac{1}{2}\sum_{f=1}^{k}\left[\left(\sum_{j=1}^{d} v_{j,f}\hat{x}_j\right)^2 - \left(\sum_{j=1}^{d} v_{j,f}x_j\right)^2\right] \\
&= \sum_{j=1}^{d} w_j x'_j + \frac{1}{2}\sum_{f=1}^{k}\sum_{j=1}^{d} v_{j,f}^2 x_j'^2 \\
&+ \frac{1}{2}\sum_{f=1}^{k}\left[\sum_{j=1}^{d}\left(v_{j,f}\hat{x}_j + v_{j,f}x_j\right)\sum_{j=1}^{d}\left(v_{j,f}\hat{x}_j - v_{j,f}x_j\right)\right] \\
&= \sum_{j=1}^{d} w_j x'_j + \frac{1}{2}\sum_{f=1}^{k}\sum_{j=1}^{d} v_{j,f}^2 x_j'^2 \\
&+ \frac{1}{2}\sum_{f=1}^{k}\left(2\sum_{j=1}^{d} v_{j,f}x_j + \sum_{j=1}^{d} v_{j,f}x'_j\right)\left(\sum_{j=1}^{d} v_{j,f}x'_j\right) \\
&= \sum_{j=1}^{d} w_j x'_j + \underbrace{\sum_{f=1}^{k}\sum_{i=1}^{d}\sum_{j=1}^{d} v_{i,f}v_{j,f}x_i x'_j}_{\text{Feature interactions between } \boldsymbol{x} \text{ and } \boldsymbol{x}'} \\
&+ \underbrace{\frac{1}{2}\sum_{f=1}^{k}\left(\sum_{j=1}^{d} v_{j,f}x'_j\right)^2 + \frac{1}{2}\sum_{f=1}^{k}\sum_{j=1}^{d} v_{j,f}^2 x_j'^2}_{\text{Feature interactions of } \boldsymbol{x}' \text{ itself}}
\end{aligned}
\tag{5}
$$

3 METHODOLOGY

3.1 Non-robust Certificates

In this section, to compute the non-robust certificate efficiently, we design a greedy approximation for the worst-case perturbation on factorization machines. Based on the certificate, we empirically demonstrate that the traditional FM model is vulnerable to discrete adversarial perturbations.

3.1.1 Greedy Solution. Inspired by the recent works [37] on adversarial attacks on graph neural networks, we adopt a greedy algorithm to compute the worst-case of δ. Note that in the works of adversarial attacks [10, 37], they usually assume that the attacker does not know the model parameters or only knows a part of the information. In this work, the goal is to develop a robustness analyzer for FMs which stands on a different perspective of previous works. Thus it is reasonable to assume we are aware of all the model parameters.

The core idea of the greedy solution is to sequentially modify the most promising components following a local optimal strategy. Since only one component is modified each time, the value of feature interactions of the perturbation vector itself is its self-interaction term $\sum_{f=1}^{k} v_{j,f}^2$. Then we need to consider the first-order weight of $\boldsymbol{x}'$ and the feature interactions between $\boldsymbol{x}$ and $\boldsymbol{x}'$ according to equation (5). Formally, given the current state of the target instance $\boldsymbol{x}^{(t)}$, we compute:

$$
j^{(t)} = \begin{cases} \arg\min\limits_{j} w_j + \sum\limits_{f=1}^{k}\sum\limits_{i=1}^{d} v_{i,f}v_{j,f}x_i + \sum\limits_{f=1}^{k} v_{j,f}^2 & f_\theta(\boldsymbol{x}) > 0 \\ \arg\max\limits_{j} w_j + \sum\limits_{f=1}^{k}\sum\limits_{i=1}^{d} v_{i,f}v_{j,f}x_i + \sum\limits_{f=1}^{k} v_{j,f}^2 & f_\theta(\boldsymbol{x}) \le 0 \end{cases},
\tag{6}
$$

where $j^{(t)}$ denotes the selected index based on the current state $\boldsymbol{x}^{(t)}$. Then the next state is obtained by flipping the component of index $j^{(t)}$. Each time a zero component is changed to one until the label of the modified instance is changed or the perturbation budget is achieved. The pseudo-code is shown in Algorithm 1. As we can see, to compute $j^{(t)}$, the time complexity is $O(n)$ where n is the number of items. The greedy algorithm requires at most q times computation of equation (6). Thus, the time complexity of the non-robust certificate is at most $O(qn)$.

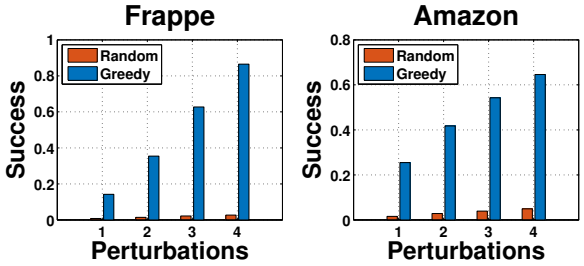

Figure 2: Impact of applying adversarial and random perturbations to the parameters of FMs on Frappe and Amazon datasets. The greedy method outperforms the random method. When the perturbation budget is 3, over half of the instances are non-robust.

3.1.2 Results. Figure 2 demonstrates the impact of applying adversarial and random perturbations to FMs with different perturbation budget q (ranging from 1 to 4) on our experimental datasets (details see Section 4.1). Specifically, we compare the attack success rate on the testing set between our greedy and the random algorithm which randomly injects user-item interactions into instances. An instance is successfully attacked if its predicted label is changed.

According to the results, all datasets show that the greedy method achieves a more significant attack performance than the random perturbations. For example on Frappe, when adding $q = 2$ user-item interactions, applying random perturbations only successfully change 1.44% instances' predictions; in contrast, the greedy method successfully changes 35.44% instances' predictions – 25 times larger than that of random perturbations. Moreover, when the perturbation number is 3, over 50% instances on Frappe (62.71%) and Amazon (54.25%) are certifiably non-robust via our method.

These results indicate that the current FM is rather vulnerable to perturbations on users' historical interactions. If a user interacts (e.g clicks) items that does not reflect his/her preference, most of the

Algorithm 1: Non-robust Certificate

Input: Model parameters f_θ, target instance $\boldsymbol{x}$, perturbation budget q;
Output: Perturbed feature $\hat{\boldsymbol{x}}$;
$t \leftarrow 0$;
$\boldsymbol{x}^{(0)} \leftarrow \boldsymbol{x}$;
while $t < q$ **do**
  Compute $j^{(t)}$ according to the equation (6);
  Obtain $\boldsymbol{x}^{(t+1)}$ via flipping $j^{(t)}$ component of $\boldsymbol{x}^{(t)}$;
  if $sign(f_\theta(\boldsymbol{x}^{(t+1)})) \neq sign(f_\theta(\boldsymbol{x}))$ **then**
    break;
  $t \leftarrow t + 1$;
Return $\boldsymbol{x}^{(t)}$;

predicted results may be changed. Note that the greedy algorithm outputs an approximated worst-case. This implies that there may exist non-robust instances that can not be certificated which makes the situation worse. The existence of such effective perturbations motivates us to develop robust factorization machines that are insensitive to discrete adversarial perturbations.

3.2 Robust Certificates

To certificate the robustness of an instance whose predicted label is -1, we derive an upper bound of δ reached by all possible perturbations. Similarly, for the instance whose predicted label is 1, the lower bound of δ is derived. If the bound is certifiably robust, no perturbations can change the prediction of the instance. To compute the bound, the calculation of equation (5) is divided into two sub-problems. To better illustrate our method, we consider the case of the predicted label is -1. Note that in the other case (i.e. the predicted label is 1), we only need to replace the following max operation (equation (7) and (8)) to the min operation. The first sub-problem is defined as:

$$b_1(\boldsymbol{x}) = \max_{\boldsymbol{x}' \in \mathcal{P}_q(\boldsymbol{x})} \sum_{j=1}^{d} w_j x'_j + \sum_{f=1}^{k} \sum_{i=1}^{d} \sum_{j=1}^{d} v_{i,f} v_{j,f} x_i x'_j \tag{7}$$

and the second sub-problem is defined as:

$$b_2(\boldsymbol{x}) = \max_{\boldsymbol{x}' \in \mathcal{P}_q(\boldsymbol{x})} \frac{1}{2} \sum_{f=1}^{k} \left(\sum_{j=1}^{d} v_{j,f} x'_j \right)^2 + \frac{1}{2} \sum_{f=1}^{k} \sum_{j=1}^{d} v_{j,f}^2 x'^2_j \tag{8}$$

The bound $b(\boldsymbol{x}) = b_1(\boldsymbol{x}) + b_2(\boldsymbol{x})$ reached by all possible perturbations is the sum of the two sub-problem's solution. If the bound does not change the predicted result, the instance x is provably robust. Although the solutions of the first and second sub-problem may be different which means such a prediction may not exist, the robustness of the instance can be verified since the bound is worse than the worst-case can be achieved. In the following subsections, the solutions of the sub-problems are elaborated.

3.2.1 The solution of the first sub-problem. The perturbation vector $\boldsymbol{x}'$ is obtained by flipping q components to 1. According to the equation (7), the impact p_j of each component x'_j to $b_1(\boldsymbol{x})$ (i.e. changing x'_j from 0 to 1) is independent which can be computed as

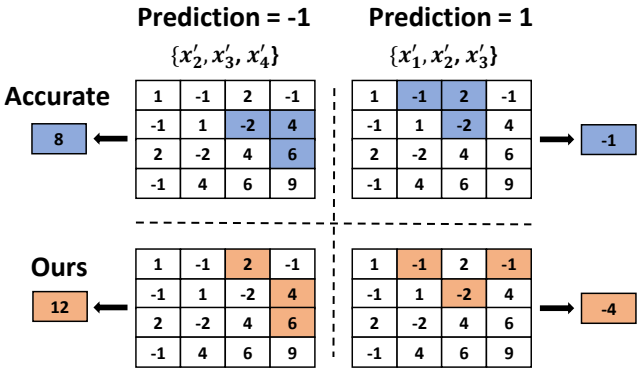

Figure 3: A running example of the solution for the second sub-problem when the predicted labels are -1 and 1. The self-interaction terms are omitted. The perturbation budget is 3 and the colored elements are selected. According to the factorization machine, only elements in the upper triangle are considered. The index sets of accurate solution (i.e. $\{x'_2, x'_3, x'_4\}$ and $\{x'_1, x'_2, x'_3\}$) are optimal.

follows:

$$p_j = w_j + \sum_{f=1}^{k} \sum_{i=1}^{d} v_{i,f} v_{j,f} x_i \tag{9}$$

Note that x_i is the original instance and it is known. It can be observed that the p_j's computation of each component x'_j is independent. Therefore, the optimal perturbation vector is obtained via flipping q components corresponding to the largest q values of $\{p_j\}_{j=1}^n$ if the predicted label is -1. Similarly, if the predicted label of the target instance is 1, the q components corresponding to the smallest q values of $\{p_j\}_{j=1}^n$ are flipped. Since some components of instances are already 1, only the zero components can be flipped.

3.2.2 The solution of the second sub-problem. Since the second sub-problem is the NP-Complete problem, an approximate solution is necessary to develop. To solve this problem, we derive the bound for the solution of the sub-problem. We first expand the quadratic term of equation (8) as follows:

$$\frac{1}{2} \sum_{f=1}^{k} \left(\sum_{j=1}^{d} v_{j,f} x'_j \right)^2 = \frac{1}{2} \sum_{f=1}^{k} \sum_{i=1}^{d} \sum_{j=1}^{d} v_{i,f} v_{j,f} x'_i x'_j, \tag{10}$$

where $x'_i x'_j$ is the interaction between the i-th and j-th component of $\boldsymbol{x}'$ and their weight is $\frac{1}{2} \sum_{f=1}^{k} v_{i,f} v_{j,f}$. Similarly, the second term of equation (8) can be view as the interaction of j-th component itself and its weight is $\frac{1}{2} \sum_{f=1}^{k} v_{j,f}^2$. Note that the weight of $x'_i x'_j$ and $x'_j x'_i$ is equal, the equation (8) can be reformulated as:

$$\begin{aligned} &\frac{1}{2} \sum_{f=1}^{k} \sum_{i=1}^{d} \sum_{j=1}^{d} v_{i,f} v_{j,f} x'_i x'_j + \frac{1}{2} \sum_{f=1}^{k} \sum_{j=1}^{d} v_{j,f}^2 x'^2_j \\ &= \sum_{i=1}^{n} \sum_{j=i}^{d} < \boldsymbol{v}_i, \boldsymbol{v}_j > x'_i x'_j, \end{aligned} \tag{11}$$

where the inner product $< \boldsymbol{v}_i, \boldsymbol{v}_j >$ denotes the weight of $x'_i x'_j$. Observed that the coefficient $\sum_{i=1}^{n} \sum_{j=i}^{n} < \boldsymbol{v}_i, \boldsymbol{v}_j >$ is actually the second-order interaction weight of the FM. Thus the second sub-problem can be view as: Given the FM's weight matrix of feature interactions and the perturbation budget q, selecting $\frac{q(q+1)}{2}$ elements which sum to the maximum (minimum). What makes the problem complicated is that the components x'_i and x'_j are dependent. Only both the x'_i and x'_j is 1, their weights $\frac{1}{2} \sum_{f=1}^{k} v_{i,f} v_{j,f}$ are taken into account. To avoid enumerating all possible perturbations, we relax this constraint. Therefore, the problem after relaxation is to select the largest (smallest) $\frac{q(q+1)}{2}$ elements from the interaction matrix without considering whether both the x'_i and x'_j are 1.

Figure 3 is a simple running example of the comparison between our method and the optimal results when the predicted labels of the instance are -1 and 1. As can be seen from the figure, when the predicted label is -1, we obtain an upper bound for the optimal solution. Analogously, when the predicted label is 1, we obtain a lower bound for the optimal solution.

3.2.3 Complexity analysis. Similar to the computation of non-robust certificate, given the perturbation budget q, the time complexity of the first sub-problem is $O(n)$. For the solution of the second sub-problem, we need to enumerate the upper triangle of the interaction matrix which has $\frac{n(n+1)}{2}$ elements, resulting in $O(n^2)$ time complexity. Thus, the overall time complexity of the robust certificates is $O(n^2)$.

3.3 Robust Training

So far, a fundamental task of certificating the FM's robustness has been addressed which is crucial to trust the model's output in real-world applications. Another important task is to train classifiers that are certifiably robust to adversarial perturbations. In this section, we show how to employ the robust certificate described above to train a robust FM.

For the training of the original FM, we use the loss following the work [24]:

$$\min_{\theta} \sum_{s \in \mathcal{D}} \log \left(1 + \exp(-y_s f_\theta(\boldsymbol{x}_s))\right), \tag{12}$$

where $\mathcal{D}$ is the set of all labeled instances. y_s is the label of the instance $\boldsymbol{x}_s$. To improve the robustness of the model, the model is enforced to predict the same results even under perturbations. To achieve this, we optimize the model to minimize the above loss under the worst-case prediction following the previous work [32]. Formally, the loss of robust training is defined as:

$$\min_{\theta} \sum_{s \in \mathcal{D}} \log \left(1 + \exp(-y_s (f_\theta(\boldsymbol{x}_s) + b(\boldsymbol{x}_s)))\right), \tag{13}$$

where $f_\theta(\boldsymbol{x}_s) + b(\boldsymbol{x}_s)$ is the bound of prediction under the worst-case perturbation. The perturbation space is $\mathcal{P}_q(\boldsymbol{x}_s)$ where q is a hyper-parameter needed to be tuned. To perform robust training, the model is first trained according to the equation (12) until convergence. Then we train the model using the equation (13) until convergence.

In practice, mini-batch training is employed to learn the model parameters. In each training epoch, a batch of training instances are randomly sampled. Then the robust certificates of each training instances are obtained (if employ robust training). Lastly, we compute the loss and use Adam algorithm [21] to optimize the model.

Table 1: Dataset statistics.

Dataset	Frappe	Amazon	Yelp
# Instance	288,609	587,373	595,191
# Feature	9,464	15,889	61,615
# User	957	6,170	16,239
# Item	4,082	2,753	14,284
# Field	11	6	7

4 EXPERIMENTS

In this section, we conduct experiments on three real-world datasets aiming to answer following research questions:

RQ1 How does our proposed robust training method perform (i.e. the accuracy and robustness) compared with state-of-the-art training approaches?

RQ2 How is the tightness of our robustness certificates? Can the robustness of most instances be certificated?

RQ3 What is the effect of hyper-parameters q in robust training?

4.1 Settings

4.1.1 Datasets. The statistics of datasets are displayed in Table 1. We briefly introduce the datasets we use as follows:

- **Frappe** [2]: Frappe is a context-aware app recommendation dataset which contains the usage logs of users under different context. The logs consist of user ID, app ID, and 8 context variables including weather, city, and daytime (e.g. morning or afternoon). Each log is converted to a feature vector using one-hot encoding and multi-hot encoding (historical interactions) resulting in 9,464 features.
- **Amazon** [16]: The amazon dataset contains product review and metadata from Amazon. Besides the user ID and item ID, the dataset contains the brand, category, and view of the items. We use multi-hot encoding to encode the category, view information, and historical interactions. The other fields are encoded by one-hot encoding which result in 15,889 features.
- **Yelp** [28]: The yelp dataset contains the user reviews on local business and attribute information of users and businesses. The dataset consists of users' interactions, social, and compliment information and businesses' city and category. We convert the business category using one-hot encoding and the other fields using multi-hot encoding which result in 61,615 features.

4.1.2 Evaluation Protocol. The datasets are randomly split into training (80%), validation (10%), and testing (10%) set. The validation set is employed to tune hyper-parameters and the performance comparison is conducted on testing set. The accuracy is reported to compare the classification performance of all models. For the comparison of the robustness, following the previous work [38], we display the average of instances' largest q that they can be certifiably robust. This metric is referred as avg-max q.

Model	factor=64						factor=32					
	Frappe		Amazon		Yelp		Frappe		Amazon		Yelp	
	Avg-max q	Acc.	Avg-max q	Acc.	Avg-max q	Acc.	Avg-max q	Acc.	Avg-max q	Acc.	Avg-max q	Acc.
FM	1.60	**93.88**	1.81	**74.62**	1.56	**80.44**	1.69	92.10	2.10	**74.23**	1.99	**80.93**
RFM-1	3.87	92.54	4.51	73.15	2.41	78.86	3.07	92.03	4.63	73.24	2.34	79.17
RFM-2	2.86	93.49	3.53	70.38	1.81	79.82	2.86	**93.49**	3.53	70.38	1.81	79.82
FM-RT	**5.26**	92.58	**7.44**	73.49	**3.36**	78.73	**5.48**	92.09	**6.76**	73.17	**3.74**	78.96
RI	+35.92%	-1.38%	+64.97%	-1.51%	+39.42%	-2.13%	+78.50%	-1.50%	+46.00%	-1.43%	+59.83%	-2.43%

Table 2: Overall performance comparison. The last row RI denotes the relative improvement over the best baseline: our robust training method significantly enhances the robustness of the FM while only has little impact on accuracy.

4.1.3 Baselines. We refer our method as **FM-RT** (robust training) and compare it with the following baselines.

- **FM** [26]: This is the benchmark factorization model that utilizes the second-order interactions between features.
- **RFM-1** [24]: This is the state-of-the-art robust factorization machine which forces the model to make right prediction under the continuous noise.
- **RFM-2** [22]: This is the factorization model whose training process is regularized by a capped l_1 norm and a capped squared trace norm.

4.1.4 Hyper-parameter Setting. We implement FM, RFM-1, RFM-2, and our model based on Pytorch, which is optimized with the Adam optimizer [21]. The batch size is fixed to 2048 for all methods and the learning rate is tuned in [0.1, 0.01, 0.001, 0.0001]. Without special mention, we show the results of embedding size 64. The uncertainty bounds η and ρ for RFM-1 are searched in [0.001, 0.002, 0.005, 0.01]. We tune capped parameters ϵ_1 and ϵ_2 for RFM-2 in the range of [0.1, 0.2, 0.5, 1] and [0.01, 0.05, 0.1, 0.5, 1] respectively. The perturbation budget q employed in our robust training is searched from 1 to 4.

4.2 Performance Comparison (RQ1)

4.2.1 Overall performance comparison. The overall performance comparison including avg-max q and accuracy of our model and other state-of-the-art methods under different embedding size are reported in Table 2. Based on the results, we have the following observations:

- FM achieves poor robustness (avg-max q less than 2 in most cases) on three datasets, indicating traditional training leads to a non-robust model which is vulnerable to discrete adversarial perturbations.
- The robustness of RFM-1 and RFM-2 are better than the FM. Although they do not consider the discrete perturbations on user-item interactions, their Avg-max q are higher than the original FM, demonstrating considering the noise in datasets can make the model less sensitive to perturbations as well.
- Jointly considering the accuracy and avg-max q, FM-RT achieves the best robustness while its accuracy is competitive to other robust training methods. For example, when the factor is 64, FM-RT improves the avg-max q over the strongest baselines by +35.92%, +64.97%, and +39.42% in Frappe, Amazon, and Yelp, respectively. Meanwhile, compared the FM (i.e. the highest accuracy among all models), it only sacrifices -1.38%, -1.51%, -2.13% of the accuracy in Frappe, Amazon, and Yelp, respectively. The improvement can be attributed to that FM-RT forces the model to predict the right label for not only the normal instances but also their worst-cases.

4.2.2 Performance comparison w.r.t the number of certificated instances. To better investigate the robustness of different approaches, the number of certifiably robust and non-robust instances under different perturbation budgets are displayed in Fig. 5 and Fig. 4 respectively. We find that:

- Compared to the robust training method, the number of certifiably robust instances of FM quickly drops to zero. For example, in Frappe, the number of instances that are certifiably robust is near zero when the perturbation number is 4. Similarly, its number of certifiably non-robust instances goes to one when the perturbation number is 5. These results indicate FM employ traditional training is non-robust.
- FM-RT consistently outperforms other methods. When the perturbation number increases, the certifiably non-robust number of compared models increases faster than FM-RT. Analogously, the certifiably robust number of compared models drops faster than FM-RT. These results demonstrate that exploiting the worst-case perturbation for training greatly facilitates the model robustness.

4.3 The Tightness of Certificates (RQ2)

The tightness of the certificate is an important metric to demonstrate how well our method approximates the worst-case perturbation. The robustness of more instances we certificate (robust or non-robust), the better our method approximates the worst-case perturbation. Towards this end, we display the number of instances that can be certificated (robust or non-robust) and can not be certificated under different perturbation budgets in Figure 6. The blue and orange area denotes the certifiably robust and non-robust instances respectively. The white area denotes the instances whose robustness can not be certificated. Based on the Fig. 6, we observe that:

As Fig. 6 shows, the robustness of most instances is certificated, demonstrating the effectiveness of our certificate. Furthermore, when the perturbation budget is small, almost all the instances are certificated. With the perturbation budget increasing, the number of instances whose robustness can not be certificated first increases then decreases. Finally, all the instances are certifiably non-robust. The reason is that the perturbation space increases rapidly when the perturbation budget enhances. Thus it becomes more difficult to

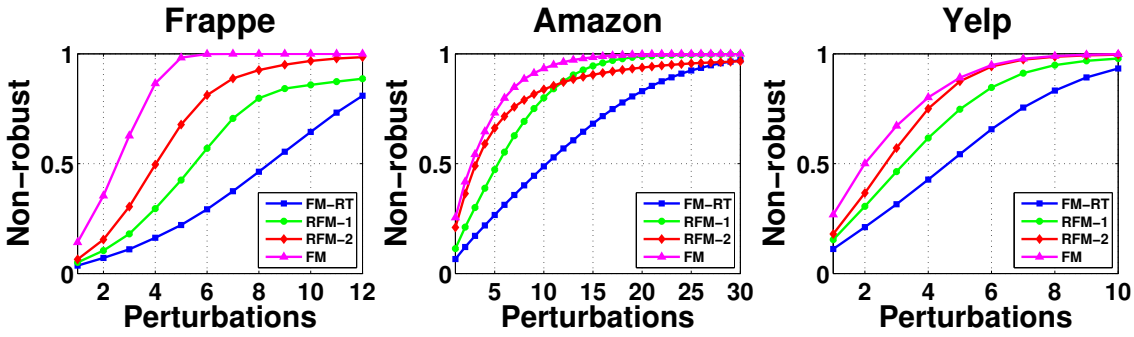

Figure 4: The number(%) of verified non-robust instances under different perturbation budgets: the number of certifiably non-robust instances after our robust training is less than the compared methods.

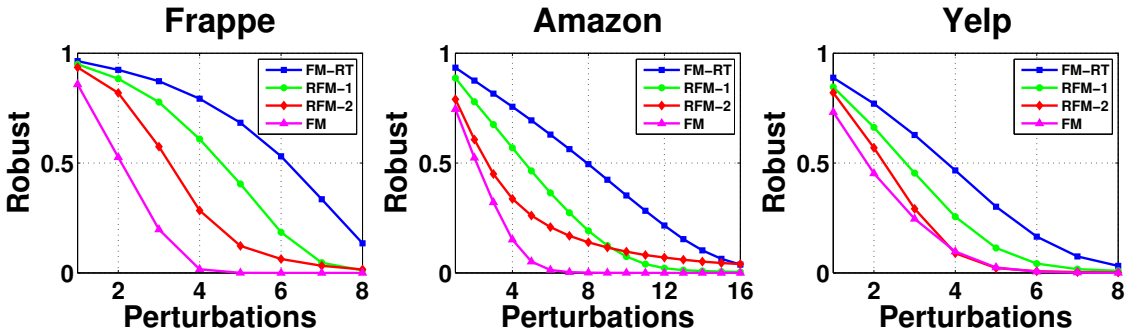

Figure 5: The number(%) of verified robust instances under different perturbation budgets: the number of certifiably robust instances after our robust training is larger than the compared methods.

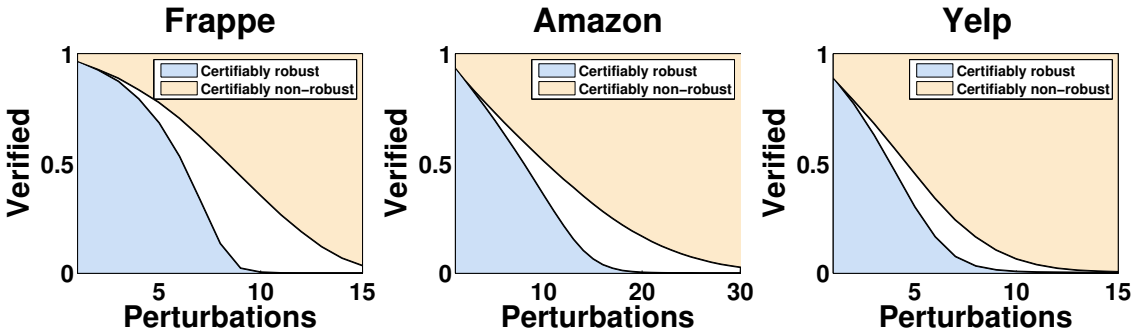

Figure 6: The number(%) of verified instances under different perturbation budgets after our robust training. The blue and orange area denotes the certifiably robust and non-robust instances respectively. The write area denotes instances that can not be certificated robust or non-robust. The robustness certificates of most instances are given.

approximate the possible perturbations set. At last, the non-robust certificate is strong enough to certificate all instances non-robust.

4.4 The Effect of Hyper-parameters(Q3)

Our method introduces a hyper-parameter q to control the worst-case perturbation during the training procedure. To investigate how the hyper-parameter q affects the model performance, we search it in the range of [1, 2, 3, 4]. Table 3 summarizes the result of avg-max q and accuracy under different training q where RT-q indicates robust training used the perturbation budget q. Furthermore, Figure 7 displays the number of robust and non-robust instances of RT-q. Jointly analyzing Table 3 and Fig. 7, we have the following observations:

- Increasing the perturbation budget q which the model is training with leads to a more robust model. As can be seen from the experimental results, under all circumstance, training used a larger

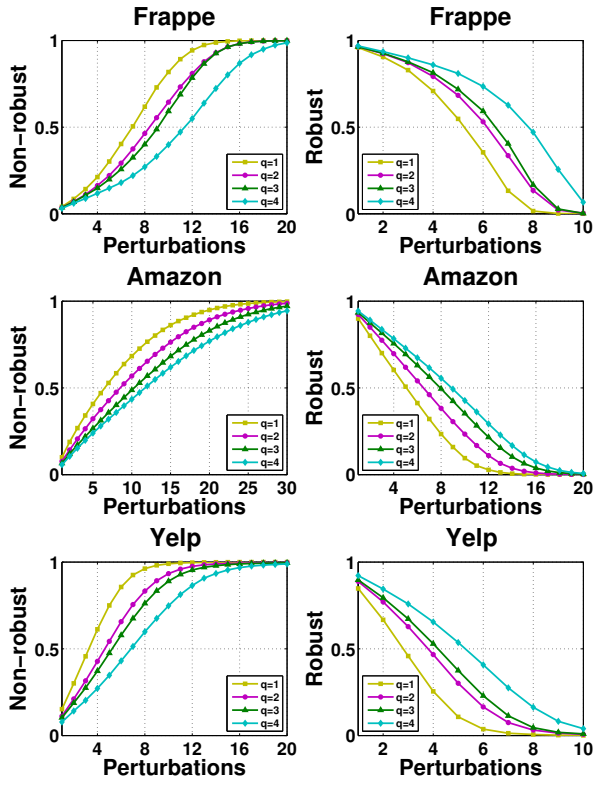

Figure 7: The number(%) of certifiably non-robust and robust instances when varying perturbation q used in our robust training.

q can achieve a larger avg-max q and increase the number of certifiably robust instances (meanwhile, the number of certifiably non-robust instances decreases). The improvement is because that during the training process, the model trained with bigger q are forced to predict the label against a stronger perturbation.

- Training used a large q will hurt the model accuracy. For example, test accuracy drops from 92.81% to 91.58% on Frappe when going from $q = 1$ to $q = 4$. The reason is that during the training process, we force the worst-case of training instances to have the same label of them (See equation (13)). The assumption may not work when the perturbation number increases. Thus, it is important to tune the hyper-parameter q to trade off accuracy and robustness.

5 RELATED WORK

In this section, we review existing work on adversarial robustness on machine learning models and robust factorization machine which are most relevant with our work.

Adversarial robustness for machine learning models. Machine learning models against adversarial perturbations have been studied extensively [13, 23]. Multiple heuristic methods [4, 11, 29, 30, 35, 36] have been proposed to improve the adversarial robustness. However, these approaches are often broken by new attack methods. Therefore, recent studies [1, 3, 6, 9, 12, 19, 25, 32, 38] have focused on the provable robustness which guarantees that no perturbation in a specific perturbation space can change an instance's prediction. The certifiable robustness has been explored to improve the robustness of various models including fully connected neural network [9], convolutional neural network [12], graph neural network [3, 38], and decision tree [1, 6]. Recently, several works [8, 14, 18, 34] have started to analyze the adversarial robustness of information retrieval models. For example, Goren et al.[14] address the robustness of learning-to-rank-based ranking functions under adversarial document manipulations. Christakopoulou and Banerjee[8] propose an adversarial attack algorithm on oblivious recommender systems via generating fake user profiles.

Dataset	Metric	RT-1	RT-2	RT-3	RT-4
Frappe	Avg-max q	4.46	5.26	5.49	6.64
	Acc.	92.82	92.58	92.09	91.58
Amazon	Avg-max q	4.84	6.21	7.44	8.28
	Acc.	74.02	73.77	73.49	73.01
Yelp	Avg-max q	2.39	3.36	3.72	4.78
	Acc.	79.40	77.73	77.60	75.41

Table 3: The performance comparison of varying perturbation q used in our robust training: when the perturbation number increases, the model robustness enhances and the accuracy decrease.

Despite great success, currently there is no robustness certificate for the factorization machine which is an important model in recommender systems. Due to the considered features of FMs are binary, the design of most robustness certificates [19, 25, 32] can not be applied. Although the work [38] has proposed to deal with the discrete feature on graph convolutional network via constructing a convex relaxation for computing a lower bound on the worst-case margin, it is inappropriate to extend it on the FM since it is non-convex.

Robust factorization machine. Recent papers [22, 24] have started to investigate perturbations on factorization machines [26]. The first robust factorization machine [24] models the perturbation via adding data uncertainty (e.g. Gaussian or Poisson perturbations) on input signals. The other method [22] considers another situation that there exist noisy training instances (i.e. the labels of input features are wrong), which is different from the problem we discuss in this paper. We show that these robust training methods are insufficient to address the discrete adversarial perturbation on instances' features. By slightly injecting user-item interactions, their prediction of most instances can be changed.

6 CONCLUSION AND FUTURE WORK

This work contributes the first work on certifying the robustness of FMs, considering the binary perturbation on instances' feature vectors. By sequentially selecting the worst-case perturbation and relaxing the dependence between features, we develop approximated solutions to certificate the robustness of FMs. We show that current FMs and the existing robust training methods are vulnerable to such perturbations. To learn a robust model, we propose a novel robust training that minimize the worst-case loss obtained by the robust certificate. Extensive experiments demonstrate that

our model is more robust. Simultaneously, the results show that our certificates are tight since the robustness certificates of most instances are given.

In future, we plan to extend our work in following directions: 1) We will jointly consider the case that flipping 0 and 1 components of feature vectors in the future. 2) we are interested in whether the same phenomenons exist when modifying other fields of features (e.g the social relations of users). 3) we will test whether the deep model such as the neural factorization machine [17], deepFM [15], or deep&cross network [31] is vulnerable to such types of perturbations and further develop a robustness analyzer for these deep models.

ACKNOWLEDGMENTS

The paper was supported by the National Natural Science Foundation of China (61702568, U1711267, U19A2079), the Guangdong Basic and Applied Basic Research Foundation (2020A1515010831), the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (2017ZT07X355), and the Key Research and Development Program of Guangdong Province of China (2018B030325001).

REFERENCES

[1] Maksym Andriushchenko and Matthias Hein. 2019. Provably robust boosted decision stumps and trees against adversarial attacks. In *NeurIPS*. 12997–13008.
[2] Linas Baltrunas, Karen Church, Alexandros Karatzoglou, and Nuria Oliver. 2015. Frappe: Understanding the Usage and Perception of Mobile App Recommendations In-The-Wild. *CoRR* abs/1505.03014 (2015).
[3] Aleksandar Bojchevski and Stephan Günnemann. 2019. Certifiable Robustness to Graph Perturbations. In *NeurIPS*. 8317–8328.
[4] Stefano Calzavara, Claudio Lucchese, and Gabriele Tolomei. 2019. Adversarial Training of Gradient-Boosted Decision Trees. In *CIKM*. 2429–2432.
[5] Hongge Chen, Huan Zhang, Duane S. Boning, and Cho-Jui Hsieh. 2019. Robust Decision Trees Against Adversarial Examples. In *ICML*. 1122–1131.
[6] Hongge Chen, Huan Zhang, Si Si, Yang Li, Duane S. Boning, and Cho-Jui Hsieh. 2019. Robustness Verification of Tree-based Models. In *NeurIPS*. 12317–12328.
[7] Liang Chen, Yang Liu, Zibin Zheng, and Philip S. Yu. 2018. Heterogeneous Neural Attentive Factorization Machine for Rating Prediction. In *CIKM*. 833–842.
[8] Konstantina Christakopoulou and Arindam Banerjee. 2019. Adversarial attacks on an oblivious recommender. In *RecSys*. 322–330.
[9] Francesco Croce, Maksym Andriushchenko, and Matthias Hein. 2019. Provable Robustness of ReLU networks via Maximization of Linear Regions. In *AISTATS*. 2057–2066.
[10] Hanjun Dai, Hui Li, Tian Tian, Xin Huang, Lin Wang, Jun Zhu, and Le Song. 2018. Adversarial Attack on Graph Structured Data. In *ICML*. 1123–1132.
[11] Fuli Feng, Xiangnan He, Jie Tang, and Tat-Seng Chua. 2019. Graph Adversarial Training: Dynamically Regularizing Based on Graph Structure. *arXiv preprint* arXiv:1902.08226 (2019).
[12] Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin T. Vechev. 2018. AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation. In *IEEE S&P*. 3–18.
[13] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. In *ICLR*.
[14] Gregory Goren, Oren Kurland, Moshe Tennenholtz, and Fiana Raiber. 2018. Ranking Robustness Under Adversarial Document Manipulations. In *SIGIR*. 395–404.
[15] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *IJCAI*. 1725–1731.
[16] Ruining He and Julian J. McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *WWW*. 507–517.
[17] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *SIGIR*. 355–364.
[18] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. 2018. Adversarial Personalized Ranking for Recommendation. In *SIGIR*. 355–364.
[19] Matthias Hein and Maksym Andriushchenko. 2017. Formal Guarantees on the Robustness of a Classifier against Adversarial Manipulation. In *NIPS*. 2266–2276.
[20] Bryan Hooi, Neil Shah, Alex Beutel, Stephan Günnemann, Leman Akoglu, Mohit Kumar, Disha Makhija, and Christos Faloutsos. 2016. BIRDNEST: Bayesian Inference for Ratings-Fraud Detection. In *SDM*. 495–503.
[21] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.
[22] Chenghao Liu, Teng Zhang, Jundong Li, Jianwen Yin, Peilin Zhao, Jianling Sun, and Steven C. H. Hoi. 2019. Robust Factorization Machine: A Doubly Capped Norms Minimization. In *SDM*. 738–746.
[23] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *ICLR*.
[24] Surabhi Punjabi and Priyanka Bhatt. 2018. Robust Factorization Machines for User Response Prediction. In *WWW*. 669–678.
[25] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. 2018. Semidefinite relaxations for certifying robustness to adversarial examples. In *NeurIPS*. 10900–10910.
[26] Steffen Rendle. 2010. Factorization Machines. In *ICDM*. 995–1000.
[27] Steffen Rendle. 2012. Factorization Machines with libFM. *ACM TIST* 3, 3 (2012), 57:1–57:22.
[28] Chuan Shi, Zhiqiang Zhang, Ping Luo, Philip S. Yu, Yading Yue, and Bin Wu. 2015. Semantic Path based Personalized Recommendation on Weighted Heterogeneous Information Networks. In *CIKM*. 453–462.
[29] Jinhui Tang, Xiaoyu Du, Xiangnan He, Fajie Yuan, Qi Tian, and Tat-Seng Chua. 2020. Adversarial Training Towards Robust Multimedia Recommender System. *IEEE Trans. Knowl. Data Eng.* 32, 5 (2020), 855–867.
[30] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian J. Goodfellow, Dan Boneh, and Patrick D. McDaniel. 2018. Ensemble Adversarial Training: Attacks and Defenses. In *ICLR*.
[31] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & Cross Network for Ad Click Predictions. In *ADKDD*. 12:1–12:7.
[32] Eric Wong and J. Zico Kolter. 2018. Provable Defenses against Adversarial Examples via the Convex Outer Adversarial Polytope. In *ICML*. 5283–5292.
[33] Fenfang Xie, Liang Chen, Yongjian Ye, Yang Liu, Zibin Zheng, and Xiaola Lin. 2018. A Weighted Meta-graph Based Approach for Mobile Application Recommendation on Heterogeneous Information Networks. In *ICSOC*. 404–420.
[34] Feng Yuan, Lina Yao, and Boualem Benatallah. 2019. Adversarial Collaborative Neural Network for Robust Recommendation. In *SIGIR*. 1065–1068.
[35] Stephan Zheng, Yang Song, Thomas Leung, and Ian J. Goodfellow. 2016. Improving the Robustness of Deep Neural Networks via Stability Training. In *CVPR*. 4480–4488.
[36] Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. 2019. Robust Graph Convolutional Networks Against Adversarial Attacks. In *KDD*. 1399–1407.
[37] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. 2018. Adversarial Attacks on Neural Networks for Graph Data. In *KDD*. 2847–2856.
[38] Daniel Zügner and Stephan Günnemann. 2019. Certifiable Robustness and Robust Training for Graph Convolutional Networks. In *KDD*. 246–256.