Applied Soft Computing 107 (2021) 107311

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Robustness analysis of classical and fuzzy decision trees under adversarial evasion attack

Patrick P.K. Chan [a], Juan Zheng [a,*], Han Liu [b], E.C.C. Tsang [c], Daniel S. Yeung [d,1]

[a] School of Computer Science and Engineering, South China University of Technology, Guangzhou, China
[b] College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China
[c] Faculty of Information Technology, Macau University of Science and Technology, Taipa, Macau
[d] SMC Society of IEEE, China

ARTICLE INFO

Article history:
Received 14 January 2020
Received in revised form 8 March 2021
Accepted 12 March 2021
Available online 27 March 2021

Keywords:
Adversarial learning
Decision tree
Fuzzy decision tree
Robustness
Evasion attack

ABSTRACT

Although decision trees have been widely applied to different security related applications, their security has not been investigated extensively in an adversarial environment. This work aims to study the robustness of classical decision tree (DT) and Fuzzy decision tree (FDT) under evasion attack that manipulate the features in order to mislead the decision of a classifier. To the best of our knowledge, existing attack methods cannot be applied to DT due to non-differentiation of its decision function. This is the first attack model designed for both DT and FDT. Our model quantifies the influence of changing a feature on the decision. The effectiveness of our method is compared with Papernot (PPNT) and Robustness Verification of Tree-based Models (RVTM), which are state-of-the-art attack methods for DT, and the attack methods employing surrogate and Generative Adversarial Network (GAN) methods. The experimental results suggest that the fuzzifying process increases the robustness of DT. Moreover, FDT with more membership functions is more vulnerable since a smaller number of features is usually used. This study fills the gap of examining the security issue of fuzzy systems in an adversarial environment.

1. Introduction

Machine learning has widespread applications, *e.g.*image recognition [1], speech recognition [2], spam [3] and malware detection [4], due to its excellent performance. However, some recent studies report that many machine learning techniques are vulnerable in an adversarial environment [5–9] in which samples are deliberately manipulated by an adversary in order to mislead the machine learning systems, for example, malware detection and spam filtering [10–12]. The performance of machine learning methods may drop significantly under an adversarial attack since the similarity assumption on the distribution of training and test samples is violated [12–14]. This vulnerability of machine learning raises public security concerns [15].

There are a number of studies focusing on the robustness of machine learning algorithms in an adversarial environment. The vulnerability is identified by proposing attack strategies to manipulate the samples in the training and test processes, *i.e.* poisoning [13] and evasion attack [12]. On the other hand, many methods have been proposed to increase the robustness of a learning algorithm against adversarial attack. For example, suspected attack samples are removed before training in data sanitization [16–18]; robust features are selected to increase the cost of attack [19]; and the robustness is considered in the learning objective in robust learning models [20,21]. A few studies find that the robustness of a model can be improved by increasing the model complexity [22,23], as well as the transferability of attack samples [24].

Although fuzzy systems are among the most popular machine learning techniques, to the best of our knowledge, the vulnerability of a fuzzy system has not been investigated in depth. This study attempts to investigate whether and how fuzzifying a system affects its robustness against evasion attack in an adversarial environment. Classical and Fuzzy Decision Trees (DT and FDT) are the focus of this paper due to their popularity in broad applications, *e.g.*multi-agent systems [25], natural language process processing [26], diagnosis systems [27], malware detection [28,29], network intrusion detection [30] and spam filtering [31]. A fair comparison on the robustness of DT and FDT requires a general evasion attack which is able to attack both DT and FDT. However, existing evasion attack methods [24,32] are specifically designed for DT and they are difficult to be extended

* Corresponding author.
E-mail addresses: patrickchan@ieee.org (P.P.K. Chan), juanzheng007@gmail.com (J. Zheng), han.liu@szu.edu.cn (H. Liu), cctsang@must.edu.mo (E.C.C. Tsang), danyeung@ieee.org (D.S. Yeung).
[1] Retired.

https://doi.org/10.1016/j.asoc.2021.107311
1568-4946/

for FDT. As a result, an evasion attack strategy which can attack both DT and FDT is first proposed in Section 3 for the discussion on the vulnerability of DT and FDT.

Furthermore, the robustness of DT and FDT against evasion attack is investigated experimentally in Section 4. Our proposed method is compared with Robustness Verification of Tree-based Models (RVTM) [32] and Papernot (PPNT) [24], which are state-of-the-art attack methods designed for DT, utilizing a surrogate model and Generative Adversarial Network (GAN). The fuzzification, *i.e.*the number of fuzzy membership functions and the robustness of FDT is also discussed. The results confirm that FDT is more robust than DT under evasion attack with the same ability. Using more fuzzy membership functions will increase the accuracy of FDT in an untainted environment, but it also reduces the robustness in general because the decision of FDT with more fuzzy membership functions consider fewer features. These findings are consistent with the ones reported in [22,23], which show that a more complicated system is usually more robust than a simple one. Finally, the conclusion and future work are discussed in Section 5.

2. Background

In this section, we first introduce the concept of adversarial attack and discuss the decision tree learning approach in dealing with uncertainty. Then we present some related works in adversarial learning and decision tree.

2.1. Adversarial evasion attack

An adversary may manipulate the data in order to mislead the decision of a machine learning system in a security related application, *e.g.*spam filtering and malware detection [3–5]. However, standard machine learning methods (implicitly) assume that the training and test data follow the same (or similar) distribution [7,8]. As the design of these methods does not consider the existence of an adversary, any slight change on the data may significantly downgrade the performance of a machine learning system [6,10–12,33].

According to the taxonomy in [34], adversarial attack can be categorized in three aspects: *attack influence*, *security violation* and *attack specificity*. *Attack influence* includes poisoning (causative) and evasion (exploratory) attack. Poisoning attack [13, 35] contaminates the training data to corrupt the learning process while the test data is manipulated to mislead the system in evasion attack [12,36–38]. Integrity, availability and privacy violation are three types of *security violation*. Integrity violation focuses on misclassifying a malicious sample as a benign one. Availability violation means reducing the overall accuracy of a system. Privacy violation refers to the leaking of protected information. *Attack specificity* defines whether an attack focuses on a target group (*i.e.*targeted attack) or not (*i.e.*indiscriminate attack).

The attack scenario discussed in this paper falls into an indiscriminate-availability-evasion attack, in which an adversary manipulates test data to evade the detection of a system. It is one of the most common adversarial attack in many applications, *e.g.*crafting adversarial samples against Deep Neural Network (DNN) [39–41] and spam filtering [12]. The evasion attack is formulated as an optimization problem [12,19] as follows:

$$\begin{aligned} &\mathbf{x}^* = argmin\ d(\mathbf{x}, \mathbf{x}') \\ &\text{s.t. } f(\mathbf{x}) \neq f(\mathbf{x}'), \end{aligned} \tag{1}$$

where d is a function which measures the distance between the original sample $\mathbf{x}$ and its attacked sample $\mathbf{x}'$, and $f : \mathbb{X} \rightarrow \mathbb{Y}$ is a classifier which outputs the class label of $\mathbf{x}$. The goal of problem (1) is to change the decision of f by crafting $\mathbf{x}$ with the minimum manipulation. Commonly used distance functions include L_1, L_2 and L_∞ [42].

Most of existing evasion attack methods target on a classifier with a continuous output, *e.g.*DNN [12,41,43], Support Vector Machine (SVM) [12,14], and Linear Discriminant classifier [12,39,44]. An adversarial sample is crafted by differentiating the output value with respect to the input [12]. These methods cannot be applied directly to a label-output classifier, *e.g.*, DT.

An attack algorithm, PPNT, is designed against DT [24]. Given the structure of the targeted DT, the attack aims to mislead the DT to classify a targeted sample to the leaf node with a different class near the node of the original prediction by modifying features of the sample. RVTM [32] proposes an attack method to a single and ensemble DT with small perturbation. However, both attack methods may not generalize from DT to FDT due to the difference between the decision strategy of DT and FDT. DT only relies on one path from the root to a leaf node in making a decision, while FDT may consider more than one path. A general attack method which uses a surrogate [45] or GAN models [46,47] is used to attack DT. This kind of methods learns a targeted classifier from its label [46] and continuous-valued outputs [48] on a number of queries. This method crafts attack samples based on a differentiable function which approximates a targeted classifier by a number of queries. However, the attack may not be efficient if the approximation is not satisfied. Another type of evasion attack method [49] does not require any knowledge of the targeted classifier, but assumes that the decision of a classifier on a sample can be queried. The attack sample is crafted by querying the revised samples iteratively. The drawback of this attack method is its time complexity due to the huge number of trials.

2.2. Classical and Fuzzy Decision Tree

DT divides a dataset into smaller disjoint subsets based on feature values iteratively until all samples in each small set belong to the same class. Since DT can be used to visually and explicitly represent decisions and decision making, it has been used widely in many applications, *e.g.*malware detection [28,29], network intrusion detection [30] and spam filtering [31]. This study chooses Iterative Dichotomiser 3 (ID3) [50], Fuzzy ID3 [27], and Yuan and Shaw's Fuzzy decision tree algorithm proposed in [51] as representatives of DT and FDT respectively due to their satisfactory performance.

2.2.1. ID3 Decision Tree

The main idea behind ID3 is to separate a set of samples into disjoint subsets according to the feature yielding the maximum information gain (IG). Each $\mathbf{x}$ in a given dataset $\mathscr{X}$ contains n features, $\mathbf{x} = \{x_1, \ldots, x_n\}$. The feature x_j contains e linguistic values, *i.e.*$S(x_j) = \{x_j^1, \ldots, x_j^e\}$. $\mathbf{x}$ belongs to the class y, where $y = 1, 2, \ldots, c$ and c is the total number of classes. IG of all candidate features are calculated by (2).

$$I(\mathscr{X}, x_j) = E(\mathscr{X}) - \sum_{v=1}^{t} \frac{|\mathscr{X}_{x_j^v}|}{|\mathscr{X}|} \cdot E(\mathscr{X}_{x_j^v}) \tag{2}$$

where $\mathscr{X}_{x_j^v}$ *in*$\mathscr{X}$ represents a set containing samples in which x_j is equal to x_j^v, $|\cdot|$ denotes the cardinality of a set, and $E(\mathscr{X})$ denotes the entropy defined as follows:

$$E(\mathscr{X}) = \sum_{k=1}^{c} -P_k \cdot log_b P_k \tag{3}$$

where P_k is the proportion of samples in the class k, b denotes the base number of the logarithmic function. $b = 2$ is commonly used in information theory. If x_j is continuous-valued, IG can be calculated after discretization [52].

$\mathscr{X}$ is divided into smaller disjoint subsets according to different values of the feature with the maximum IG. This procedure is applied to each subset again until all samples in each set belong to a single class or all features have been examined.

A sample is classified through a top-down traversal of the decision tree starting at the root node. A branch is chosen at each node according to its feature values until reaching a leaf node representing a class to which the sample belongs. ID3 has been applied to many applications, *e.g.*Fraud detection [53] and Network Intrusion Detection [54], due to its superior performance and ease of interpretation.

2.2.2. Fuzzy decision tree

In reality, a concept may not be crisp, *e.g.*the vagueness and ambiguity in human thinking and perception [51]. From this point of view, the concept of partial truth cannot be represented by Boolean logic. Fuzzy logic is the extension of Boolean logic, which represents a continuous truth value between 0 and 1. Fuzzy DT is one of the applications of fuzzy logic, which enables DT to deal with imprecise knowledge [55]. Given a fuzzy dataset, each linguistic term x_j^v is represented by a fuzzy set $\mathscr{X}_{x_j^v} = (\mathscr{X}, \mu_{x_j^v})$, where $\mu_{x_j^v}(\mathbf{x})$ is a membership value indicating the degree of x_j to x_j^v. A crisp dataset should first be fuzzified according to a membership function, such as trapezoid-shaped or triangle-shaped functions, before using FDT.

A number of FDT algorithms can be found in the literature. Two heuristics for constructing FDTs similar to ID3 are adopted to have a fair comparison with DT in this study. Fuzzy ID3 (FID3) [27] is considered since IG is used as the feature selection criterion in both FID3 and ID3. On the other hand, the FDT proposed by Yuan and Shaw (YST) [51] only classifies a sample based on one decision rule, which is similar to ID3.

FID3 [27] is extended from ID3 [50]. Same as ID3, IG, defined as (2), is used as the criterion of feature selection. The cardinality of the fuzzy set, $\mathscr{X}_{x_j^v}$ is computed by (4).

$$|\mathscr{X}_{x_j^v}| = \sum_{\mathbf{x}\in\mathscr{X}} \mu_{x_j^v}(\mathbf{x}) \tag{4}$$

In contrast with ID3, the leaf node of FID3 represents each class with the possibility calculated by the proportion of data with respect to the class. When classifying a test sample, multiple paths from the root to each leaf node are considered. A node and its branch represent x_j and its membership function. For each path from the root to a left node, the output product of all fuzzy membership functions on the path multiplies with the class possibility vector of the leaf node. The class with the largest summation of membership values is chosen for classifying the test sample, *i.e.*the test sample is classified into the class with the highest overall membership value.

The average ambiguity is used as the feature selection criterion in YST. For feature x_j, the average ambiguity is the weight of the ambiguity of its linguistic values:

$$G(x_j) = \sum_{v=1}^{t} P_{jv} \cdot g_{x_j^v} \tag{5}$$

where P_{jv} represents the proportion of samples whose feature x_j is x_j^v, and $g_{x_j^v}$ denotes the ambiguity of x_j^v defined as follows:

$$g_{x_j^v} = \sum_{k=1}^{c} (\pi_{jv}^k - \pi_{jv}^{k+1}) \cdot \ln k \tag{6}$$

where $\pi_{jv}^k = \frac{P_{jv}^k}{\max_{r\in 1,2,\ldots,c} P_{jv}^r}$ measures the possibility of classifying a sample to the class k under the condition of $x_j = x_j^v$. P_{jv}^k denotes the proportion of samples belonging to class k. $\pi_{jv}^{c+1} = 0$ and $\pi_{jv}^s \geq \pi_{jv}^w$, where $s < w$.

The class of an unseen sample is identified by traversing from the root node to each leaf node of a fuzzy DT in order to calculate the membership values of all the nodes on the path. The minimum membership value along the path is selected for a class. The class with the highest membership degree is chosen to label the unseen sample.

3. Proposed evasion attack model against classical and Fuzzy Decision Trees

The decision of DT only relies on one path between the root and leaf nodes, which is different from a FDT. The attack methods [24,45] for DT may not be applicable to a FDT. Moreover, these attack methods are mainly designed for Boolean values. Therefore, in order to have a fair comparison on the robustness of DT with FDT in an adversarial environment, a general model of evasion attack applicable to both DT and FDT is required. This section proposes a general evasion attack model against both DT and FDT. The adversarial setting, framework, and time complexity of our proposed attack will be discussed.

3.1. Adversarial setting

The adversarial setting of our proposed attack is discussed according to the framework presented in [34], including the adversarial goal, knowledge and capability.

The *goal* of our attack is to evade the detection of the targeted classifiers, *i.e.*DT and FDT, on an attack sample. The *capability* of an adversary is to manipulate any number of feature values of the attack sample. Modification on an attack sample is bounded by d_{max} the maximum distance between the original sample and its attack sample. *Perfect* and *limited knowledge* are considered: all the information, including the feature space, structure and all parameters of the target classifier, is obtained in *perfect knowledge* (PK), while only the feature space is known in the setting of *limited knowledge* (LK). A surrogate model is trained by using samples with the label information queried from the target model.

3.2. A greedy approach based Evasion Attack framework for Decision Tree

A general attack model framework for DT and FDT, is first devised in this section. The detailed implementation on DT and FDT is then introduced in sub-sections. As DT is non-differentiable, no gradient information can be obtained guide the attack manipulation. Another criterion which indicates the influence of changing a feature on successful attack is proposed. We define M as the impact of manipulating $\mathbf{x}$ on the confidence of a targeted sample $\mathbf{x}$ for its true class t, which is defined as follows:

$$M(\mathbf{x}, \Delta\mathbf{x}_f) = C^t(\mathbf{x}) - C^t(\mathbf{x} + \Delta\mathbf{x}_f), \tag{7}$$

where $C^t(\mathbf{x})$ is the output confidence on $\mathbf{x}$ for the class t, and $\Delta\mathbf{x}_f = [\Delta x_1, \Delta x_2, \ldots, \Delta x_n]$ represents the feature manipulation. When $f \neq f$, $\Delta x_f = 0$; otherwise, $\Delta x_f = \epsilon$. The value of ϵ can be set according to the domain of a feature in an application. In our model, both positive and negative $\Delta\mathbf{x}_f$ are considered, *i.e.*$M(\mathbf{x}, +\Delta\mathbf{x}_f)$ and $M(\mathbf{x}, -\Delta\mathbf{x}_f)$. An attack sample ($\mathbf{x}'$) is then crafted from the original $\mathbf{x}$ iteratively. In each iteration, the feature with the largest $M(\mathbf{x}, +\Delta\mathbf{x}_f)$ and $M(\mathbf{x}, -\Delta\mathbf{x}_f)$ is chosen. The process repeats until the sample is misclassified, or the total manipulation exceeds the maximum distortion, or the iteration time is more than the maximum number. The detailed algorithm is shown in Algorithm 1. C^t is defined and discussed for different kinds of decision trees in the following sub-sections.

Algorithm 1 Evasion Attack Framework for DT

Input: $\mathbf{x}$: the original sample; t: the true class of $\mathbf{x}$; ϵ: the perturbation of a feature; d_{max}: the maximum attack strength; i_{max}: the maximum number of loops; g: the decision function of DT.
Output: $\mathbf{x}'$: the manipulated sample

$\mathbf{x}' \leftarrow \mathbf{x}$
$i \leftarrow 0$
repeat
  $i \leftarrow i + 1$
  $f^{+^*} \leftarrow \arg\max_{f \in \mathbf{x}'}(M(\mathbf{x}', +\Delta\mathbf{x}_f))$
  $f^{-^*} \leftarrow \arg\max_{f \in \mathbf{x}'}(M(\mathbf{x}', -\Delta\mathbf{x}_f))$
  if $M(\mathbf{x}', +\Delta\mathbf{x}_{f^{+*}}) > M(\mathbf{x}', -\Delta\mathbf{x}_{f^{-*}})$ **then**
    $\mathbf{x}'_{f^{+*}} \leftarrow \mathbf{x}'_{f^{+*}} + \epsilon$
  else
    $\mathbf{x}'_{f^{-*}} \leftarrow \mathbf{x}'_{f^{-*}} - \epsilon$
  end if
  $t' = g(\mathbf{x}')$
until $i > i_{max}$ or $d(\mathbf{x}', \mathbf{x}) > d_{max}$ or $t' \neq t$
return $\mathbf{x}'$

3.2.1. Implementation of Evasion Attack against DT

$\mathscr{H}$ is defined as a set containing all decision paths from the root node to leaf nodes in DT. We let the function $L(h)$ be the number of internal nodes in h, while the function $D(h, \mathbf{x})$ denotes the number of internal nodes in which the condition is true for $\mathbf{x}$. In ID3, a sample $\mathbf{x}$ is assigned to a class associated with a path h^* if all conditions in h^* are true for $\mathbf{x}$, *i.e.*$D(h^*, \mathbf{x}) = L(h^*)$. Therefore, reducing $D(h^*, \mathbf{x})$ will reduce the chance of assigning $\mathbf{x}$ to its true class. C is defined as follows for ID3:

$$C^t_{ID3}(\mathbf{x}) = \max_{h \in \mathscr{H}^t} \frac{D(h, \mathbf{x})}{L(h)} \tag{8}$$

where $h \in \mathscr{H}$, and $\mathscr{H}^t$ denotes a set containing all the paths with the true class t of the original sample. Since there may be more than one path that outputs t in ID3, the maximum value is considered. The attack aims to reduce the association between $\mathbf{x}$ and t. When $D(h, \mathbf{x})/L(h)$ is smaller for $h \in \mathscr{H}^t$, the chance of misclassifying $\mathbf{x}$ is higher.

3.2.2. Implementation of Evasion Attack against FDT

The confidence C is discussed for two kinds of FDTs, including two popular methods (*i.e.*FID3 [27] and YST [51]). C is designed according to the decision mechanism (*i.e.*single or multiple paths) and the definition of the confidence value of a FDT.

In FID3, the class with the largest summation of the product of the membership value of h ($m(h, \mathbf{x})$) and the class probability vector of its leaf node ($l(h, \mathbf{x})$) for all paths is assigned to $\mathbf{x}$. Therefore, we define C for FID3 as follows:

$$C^t_{FID3}(\mathbf{x}) = \sum_{h \in \mathscr{H}} m(h, \mathbf{x}) \cdot l^t(h, \mathbf{x}) \tag{9}$$

where l^t is the classification confidence of the leaf node of h on the class t for $\mathbf{x}$. However, as YST only considers the path with the highest membership value, its C is defined as:

$$C^t_{YST}(\mathbf{x}) = \max_{h \in \mathscr{H}^t} m(h, \mathbf{x}) \tag{10}$$

3.3. Time complexity

The time complexity of our attack method is $O(i_{max} \times n \times q)$ for attacking a sample, where n denotes the number of features, q denotes the time to query the targeted tree, and i_{max} is the maximum iteration number. n depends on an application, while i_{max} relates to the attack cost. Both of these factors are independent to DT. The only difference between the time complexity of evasion attack against DT and FDT is q. q is larger when DT is more complex. In general, q of DT is smaller than the one of FDT since the decision process of DT is usually simpler.

4. Experiment

The investigation on whether and how fuzzifying DT increases its robustness in an adversarial environment is discussed experimentally in this section. Section 4.1 provides the detail of the experimental setting. The proposed attack method is compared with PPNT and RVTM, which are the state-of-the-art methods aiming at DT, and the surrogate methods experimentally in Sections 4.2 and 4.3 respectively. Section 4.4 evaluates the vulnerability of DT and FDT, while the influence of the number fuzzy membership functions on the robustness is discussed in Section 4.5. Finally, the running time of the attack methods is discussed in Section 4.6.

4.1. Setting and dataset

Four datasets collected from security-related applications, including PDF malware detection (PDF), network intrusion detection (IDS), and two from spam filtering (spam), are considered in our study. 1000 samples are chosen chronologically in the PDF malware detection dataset [56]. A sample contains 114 features representing the number of occurrences of given keywords of a PDF file. A feature is normalized to [0, 1] by dividing it by 100. Two spam filtering datasets are used. The benchmark dataset TREC 2007 emails corpus [19] contains 25,220 and 50,199 legitimate and spam emails respectively. Each email is represented by a feature vector which denotes the number of occurrences of the dictionary words in an email. Words are extracted from the first 2000 emails in a chronological order. In order to keep the computational complexity manageable, the dimensionality of the feature set is reduced from more than 20000 to 200 by a supervised feature selection method based on information gain [57]. A feature is normalized to [0, 1] by TF-IDF [58], which indicates the importance of a word among all emails. Another spam filtering dataset is Spambase [59], which contains 1800 legitimate and 2800 spam mails. Each email is represented by 57 features. 54 out of 57 features are real numbers which denote the percentage of a particular word used in an email. The other three features represent the average, the longest and the sum of the length of the sequences of consecutive capital letters. The binary NSL-KDD [60] repository, which is one of the most popular intrusion detection datasets, is also considered. The training and test sets contain 124,978 and 19,366 samples with 41 features.

For all experiments except NSL-KDD, 30% of the samples are selected as a training set, and the rest form the test set. In order to reduce the bias, the original training and test sets in NSL-KDD are unionized and randomly separated into different training and test sets with the same size as the original ones. All the experiments are independently carried out five times. The attack strength is defined as the distance between the original sample and its adversarial sample counterpart, *i.e.*$\|\mathbf{x} - \mathbf{x}'\|$. The maximum attack strength (d_{max}) is set as 0.05, 0.1, 0.15, 0.2, 0.25 and 0.3. The maximum iteration number i_{max} is set as 500 for all methods according to our preliminary study. Most samples (more than 99.9%) can be crafted normally within 500 iteration for all attack methods. The distance function determines the attack features and modification intensity. Similar to [19,56], L_1 is used in the PDF dataset, while L_2 is used in the two spam datasets and NSL-KDD. The performance of the attack methods are evaluated by the F_1 score, defined as $F_1 = 2(P \times R)/(P + R)$, where P and

Table 1
Accuracy of substitute models (DT, NN and SVM) on the prediction of labels classified by the targeted models (ID3, FID3 and YST) on test data.

Target	Substitute								
	$DT_{50\%}$	$DT_{100\%}$	DT_{PK}	$NN_{50\%}$	$NN_{100\%}$	NN_{PK}	$SVM_{50\%}$	$SVM_{100\%}$	SVM_{PK}
ID3	0.98	0.98	1	0.95	0.95	0.93	0.96	0.97	0.91
FID3	0.92	0.91	1	0.97	0.98	0.89	0.99	0.99	0.90
YST	0.92	0.92	1	0.96	0.97	0.83	0.97	0.97	0.84

DT is as the same type as the targeted DT.

R are the precision and recall. A higher F_1 score indicates more adversarial samples generated by an attack method mislead the targeted model.

Each attribute is discretized into three intervals for ID3 using the method proposed in [52], while the samples are fuzzified using triangular-shaped membership functions for FDT. To avoid overfitting, the maximal depth of the tree (dp_{max}) is set to $\lceil \log_2^n \rceil$, which is suggested by [61], where n indicates the amount of features of each sample. According to the performance of the DT, the class proportion threshold used as leaf node condition (θ_r) is set as 0.85. In addition, the fuzziness control parameter α-cut, which is suggested in [51], is set as 0.2, 0.35 for FID3 and YST according to the preliminary parameter evaluation aiming to maximize the performance of the model.

The performance of our attack model is compared with the methods specifically designed for DT, including *PPNT* [24] and *RVTM* [32], and also the generic attack methods, including GAN based attack methods (*i.e.GAN1* [46] and *GAN2* [47]), and surrogate based attack methods [12,19,41]. Three popular methods, decision tree (DT), neural network (NN) and support vector machine (SVM), are chosen as the substitute models. We that assume an adversary obtains 50% and 100% of the training set randomly without true labels. This simulates partial knowledge on the targeted system. The label information is obtained by querying the target model. To be more specific, DT uses the same type of algorithm and same parameters as the target DT. NN contains 3–5 fully connected layers with 3–7 neurons and the sigmoid function is used as the activation function. The weights have been updated in 5000 iterations with 0.1 learning rate. The Gaussian kernel with the Gamma set as 1, and the linear kernel are chosen for SVM. C is chosen from {0.001, 0.01, 0.1, 1, 10, 100}. A preliminary security evaluation is used to tune the parameters of each model aiming to minimize the error rate in each application. The gradient descent based evasion attack algorithm [12] is then applied to manipulate the samples for NN and SVM.

4.2. Attack performance comparison in perfect knowledge

This section evaluates and compares the performance of the attack methods in the perfect knowledge setting. NN and SVM are trained by using all the information in the training set, a surrogate model denoted by NN_{PK} and SVM_{PK} for Sub_{NN} and Sub_{SVM} as they cannot directly attack DTs. The gradient attack algorithm [12] is then used in Sub_{NN} and Sub_{SVM}.

The similarity between the surrogate and targeted models is firstly evaluated. The accuracy of the substitute models on the labels of test samples predicted by the targeted models is shown in Table 1. The values present the average accuracy on three datasets. A higher accuracy indicates a substitute model approximates a targeted one better. Since DT_{PK} is the same model as the target one, its accuracy is always 100%. However, the accuracy of NN_{PK}, and SVM_{PK} is between 85% and 91% because the learning algorithms of SVM and NN are different from a DT. Without the output of the target model, the decision boundaries of SVM and NN are different from the one of a DT.

Fig. 1 shows the F_1-score of the attack models with different attack strengths in the perfect knowledge setting in different datasets, including PDF, Spam, Spambase and NSL-KDD from top to bottom. For each dataset, the left, middle and right graphs represent ID3, FID3 and YST respectively. The x-axis denotes the maximum attack strength d_{max} from 0 to 0.3. The dashed and solid lines indicate F_1 score of successful attack rate of test samples for two different classes, *i.e.*the class 0 and 1. Each line represents one of the attack methods. It should be noted that *PPNT* and *RVTM* are neglected in two FDTs as they are specifically designed for DT.

In ID3, the performance of the attack methods specifically designed for DT is significantly better than the generic attack methods in all datasets. The F_1 scores of *our method*, *PPNT*, and *RVTM* increase quickly to more than 0.75, 0.6, 0.8 and 0.09 in PDF, Spam, Spambase, and NSL-KDD when a small attack strength is used, *i.e.*$d_{max} = 0.1$. In contrast, *GAN1*, *GAN2*, Sub_{NN} and Sub_{SVM} do not perform well in ID3. Under the same attack strength 0.1, the F_1 scores of *GAN1*, *GAN2*, Sub_{NN} and Sub_{SVM} are only larger than 0.00, 0.01, 0.08 and 0.32 in Spambase respectively, and nearly equal to zero in PDF, Spam, and NSL-KDD. *Our method*, *PPNT*, and *RVTM* perform similarly in PDF, Spam and Spambase. In NSL-KDD, *RVTM* achieves higher successful evasion rate than *our method* and *PPNT*.

For FDTs, the F_1 scores of *our method* on attacking FID3 do not rise dramatically until the attack strength is larger than 0.15 in all datasets. For YST, *our method* is able to mislead some class 1 samples with small attack strength, but the F_1 scores of another class still remain small in Spam and Spambase when the attack strength increases. It indicates that FID3 and YST are robust to small strength attack. Similar to ID3, due to different learning strategies of NN, SVM and FID3, both Sub_{NN} and Sub_{SVM} cannot mislead the decision of FID3 in PDF, Spam and NSL-KDD effectively.

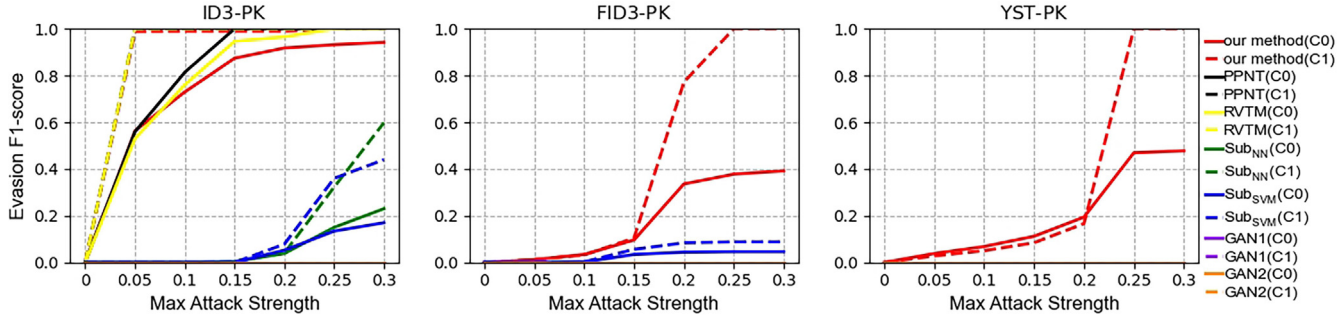
General attack performance of *GAN1*, *GAN2*, Sub_{NN} and Sub_{SVM} is not good in all cases, *i.e.*the successful evasion rate is relatively low. It may be due to the fact that the learning strategies of NN and SVM are different from ID3. This makes Sub_{NN} and Sub_{SVM} unable to attack the useful features. On the other hand, training a GAN model requires a large dataset which is not provided by the chosen applications. As a result, the attack samples generated by *GAN1* and *GAN2* perform inefficiently.

4.3. Attack performance comparison in limited knowledge

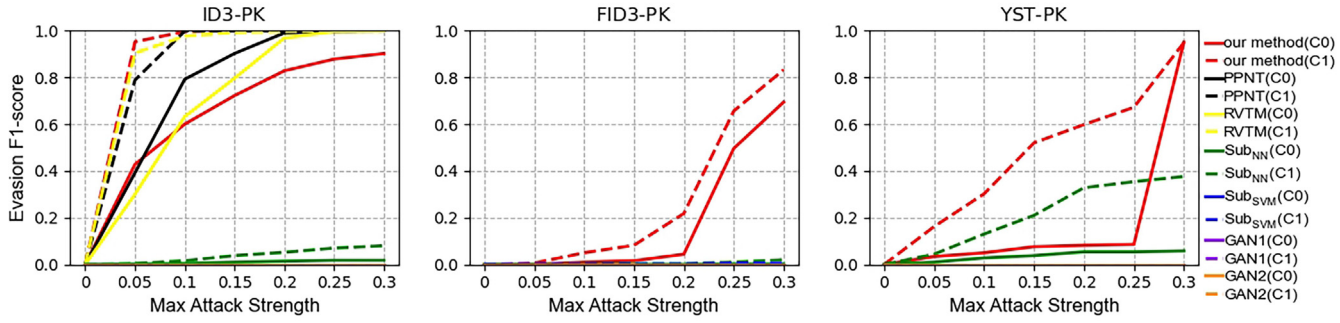
The performance of all attack methods in two limited knowledge settings, *i.e.*LK-50% and LK-100%, in which an adversary obtains 50% and 100% of the training set of the targeted model without the true labels is investigated. The label information is obtained by querying the targeted model. For Sub_{NN} and Sub_{SVM}, NN and SVM are used as a surrogate model denoted by NN_x and SVM_x, where $x = 50\%, 100\%$, while a decision tree with the same type as the targeted one is used as the surrogate model (*i.e.*$DT_{50\%}$ and $DT_{100\%}$) for *our method* and *PPNT*. The rest of the settings is the same as the ones used in the previous section.

All surrogate models with limited knowledge achieve higher than 90% accuracy, as shown in Table 1. SVM achieves the best approximation on the targeted DTs, *i.e.*the accuracy is 96% to 99%. The performance difference between NN and SVM is not apparent. It may be due to the volume of the data being not sufficiently large enough. On the other hand, DT learns ID3 the best among all methods of surrogate models.

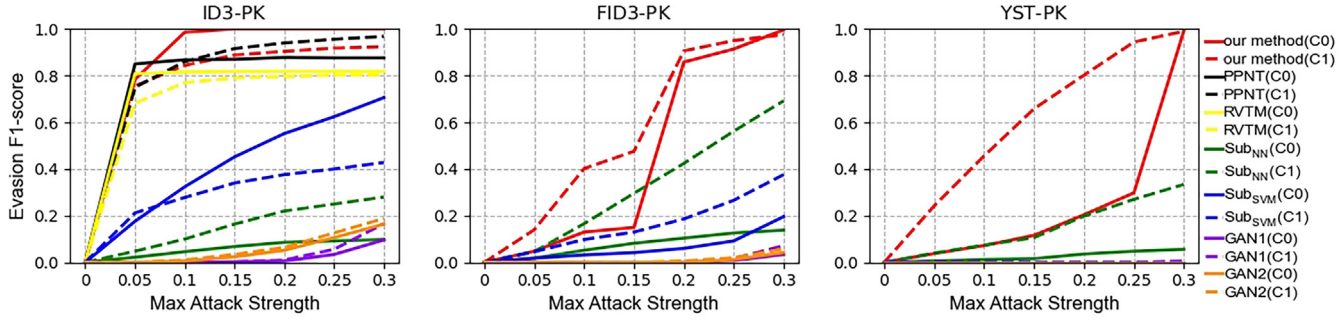
The average F_1-scores of the attack methods with different attack strengths in the limited knowledge setting with 50% and 100% training set (*i.e.*LK-50% and LK-100%) on all datasets are illustrated in Figs. 2. Similar to the previous one, *PPNT* and *RVTM* are only applied to ID3. *Our method*, *PPNT*, and *RVTM* also do not



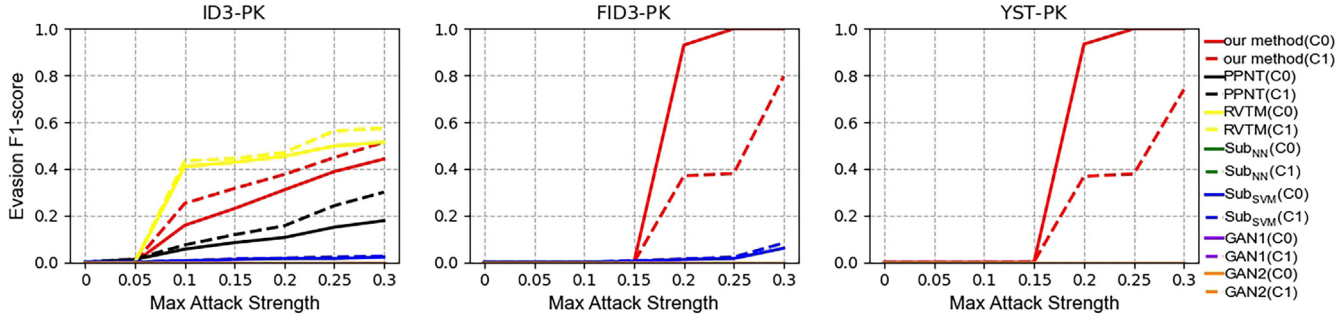
(a) PDF



(b) Spam



(c) Spambase

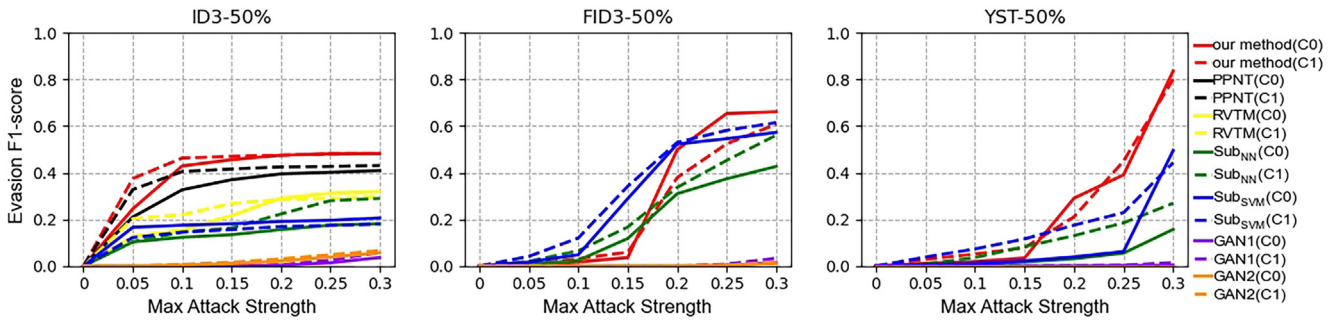


(d) NSL-KDD

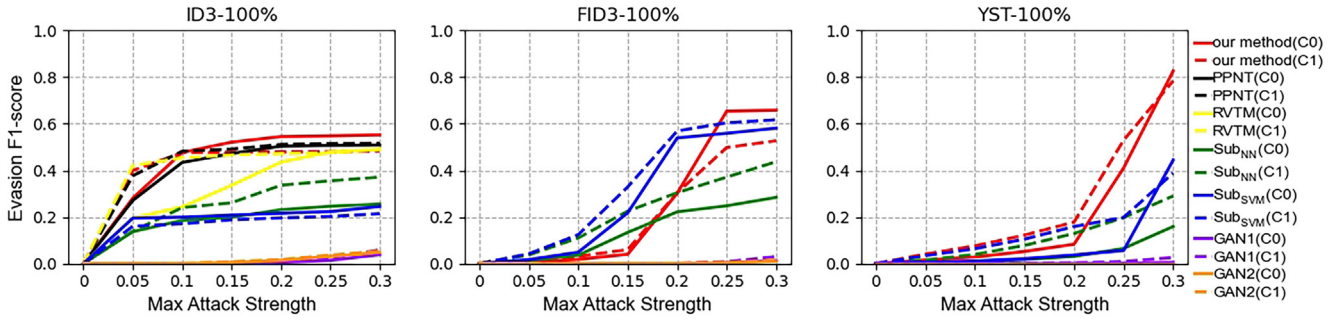
Fig. 1. Average successful attack performance of the attack methods with perfect knowledge against ID3, FID3 and YST on PDF, Spam, Spambase and NSL-KDD in terms of average F_1 score in evasion.

work efficiently in LK comparing with the ones in PK respectively. When more information of the target system is obtained (*i.e.*the perfect knowledge), these methods perform better. *RVTM* suffers from inaccurate information on the targeted classifier more than *PPNT* and *our method*, *i.e.*its F_1-scores are much lower.

Although the F_1 scores of Sub_{NN} and Sub_{SVM} are much lower than *RVTM*, *PPNT* and *our method*, the performance of Sub_{NN} and Sub_{SVM} in the limited knowledge setting is better than the ones in the perfect knowledge setting. For example, all F_1 scores of Sub_{NN} and Sub_{SVM} are around 20% when the attack strength is 0.5 in LK, but F_1 scores are nearly 0 in all datasets except Spambase in PK shown in Fig. 1. On the other hand, *GAN1* and *GAN2* still achieve nearly zero attack successful rate in all cases due to insufficient training samples.



(a) Limited knowledge with 50% training set



(b) Limited knowledge with 100% training set

Fig. 2. Average successful attack performance of the methods with limited knowledge (Querying labels from the target DT for 50% and 100% training set) against ID3, FID3 and YST on PDF, Spam, Spambase and NSL-KDD in terms of average F_1 score in evasion.

Sub_{SVM} downgrades the performance of FID3 effectively in the limited knowledge setting. For example, when the attack strength is 0.15, the F_1 scores are larger than 25% and 20% in LK-50% and LK-100% respectively. In contrast, *our method* and Sub_{NN} do not work efficiently, especially when the attack strength is small. On the other hand, YST is the most robust one among all DTs in LK. Only a few samples can be successfully attacked by all methods with the attack strength smaller than and equal to 0.15. *Our method* achieves the highest F_1 scores among all methods in YST when the attack strength is larger than 0.15. The results indicate that when more information of the target information is obtained (*i.e.*the perfect knowledge), *our method* performs better.

In general, Sub_{SVM} misleads the decision of target models more effectively than Sub_{NN} in most cases for FDTs. Moreover, the performance of Sub_{NN} and Sub_{SVM} is worse in the perfect knowledge setting when comparing them with the limited knowledge setting. The substitute model is trained by using the output of the target model in the limited knowledge setting, but the real labels are obtained in the perfect knowledge setting. As a result, NN and SVM approximate the decision boundary of the target model in the limited knowledge setting, while they learn how to separate the classes directly from the samples with perfect knowledge. As the learning strategies of NN and SVM are very different from the one of DT, the trained substitute models are different from the target ones, which is also indicated by the prediction accuracy shown in Table 1. The experimental results suggest that the more the information on the training set is obtained, the better the attack performance achieved by *our method*, *PPNT*, and *RVTM*. This is because the type of substitute models used in these methods is the same as the target model. However, the output of the target model as the label in training is important to Sub_{SVM} and Sub_{NN}.

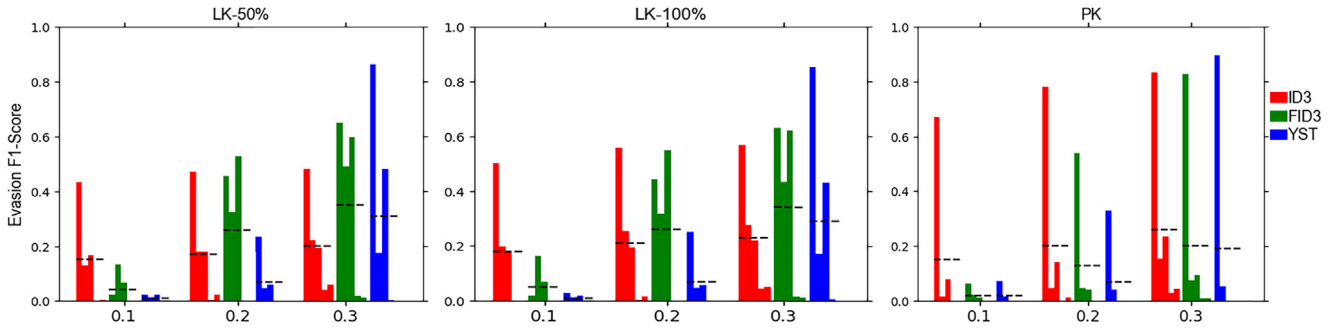
4.4. Robustness of DT and FDT against attack

The robustness of DT and FDT under *our method*, *GAN1*, *GAN2*, Sub_{NN}, and Sub_{SVM} is investigated and compared. The robustness is measured by the average F_1 scores of samples in two classes of four datasets. A larger average F_1 score indicates the model is more vulnerable to the attack. *PPNT* and *RVTM* are ignored in this section since they are not capable of attacking both DT and FDT.

Fig. 3 shows the experimental results in different knowledge settings. The *x*-axis indicates the attack strengths (0.1, 0.2 and 0.3), while the *y*-axis represents the average F_1 score. Red, green and blue represent the targeted model ID3, FID3 and YST respectively. The bars of each color group denote the attack methods from left to right, and the black dotted line represents the average value of the five attack methods.

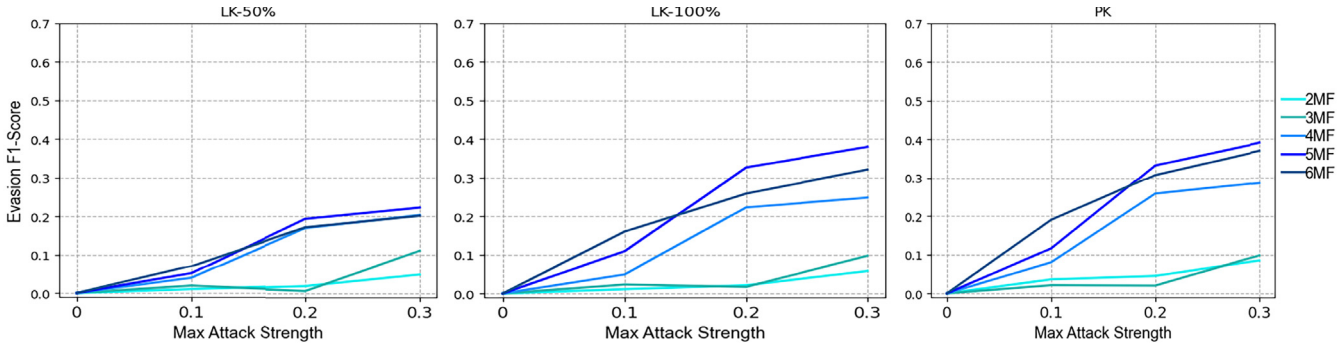
In the LK setting, the F_1 score of ID3 is much higher than FID3 and YST when the attack strength is 0.1. When the attack strength increases from 0.1 to 0.3, F_1 scores of ID3 become stable while the ones of FID3 and YST rise from 0.06 and 0.01 (0.08 and 0.01) to 0.38 and 0.51 (0.37 and 0.30) in LK-50% (LK-100%) setting respectively. The results suggest that a small modification on a sample can mislead the decision of ID3 easily, while a large modification is required for misleading FDTs. However, an attack with larger attack strength cannot reduce the accuracy of ID3. The imperfect knowledge on ID3 may cause the attack samples to be crafted ineffectively.

ID3 is the most vulnerable among all methods in PK setting, especially for *our method*, *i.e.*F_1 scores are 68%, 79%, and 83% when the strengths are 0.1, 0.2, and 0.3, respectively. Similar to the results in LK setting, FID3 and YST are robust when the attack strength is small, *i.e.*F_1 scores of both methods are less than 10% when attack strength is 0.1. When the attack strength increases, F_1 scores of all methods rise in a similar way. The difference between FID3 and YST is smaller than the ones with LK, and YST is only slightly more robust than FID3 on average.



(a) The case of $LK_{50\%}$ (b) The case of $LK_{100\%}$ (c) The case of PK

Fig. 3. Average F_1 scores of ID3, FID3 and YST under the evasion attack methods with different attack strength in limited and perfect knowledge settings.



(a) The case of $LK_{50\%}$ (b) The case of $LK_{100\%}$ (c) The case of PK

Fig. 4. Average F_1 scores of FID3 and YST under *our method*, *GAN1*, *GAN2*, Sub_{NN} and Sub_{SVM} with different strengths in perfect and limited knowledge settings.

4.5. Robustness of FDT with different numbers of fuzzy membership functions against attack

The influence of the number of fuzzy membership functions on the robustness of FDT against the attack is investigated in this section. Similar to the previous section, only *our method*, *GAN1*, *GAN2*, Sub_{NN} and Sub_{SVM} are considered. Fig. 4 shows the results in different knowledge settings. The *x*-axis denotes the number of membership functions (MF), while the *y*-axis represents the F_1 score of the successful attack. As the behavior on different numbers of MF of FID3 and YST on the attack methods is similar, only the average F_1 score of samples in two classes is shown. Different colors indicate the performance of FDT under the attack with different attack strengths.

The experimental results suggest that the robustness of FDT decreases with the increase of the number of MFs. The average F_1 scores of FDT increase when more MFs are used in most attack settings. FDT with more MFs may reduce the depth of the tree, *i.e.*fewer features are considered in decision making. Therefore, changing fewer number of features may mislead the decision more easily. F_1 scores of some FDTs are smaller in the perfect knowledge setting than the corresponding ones in the partial knowledge setting, *e.g.*F_1-scores of FDT with six MFs are 32% and 50% in PK and LK-50% settings attack strength is 0.3. The above phenomenon is likely due to the poor attack performance of Sub_{NN}, and Sub_{SVM} in the perfect knowledge setting.

4.6. Running time

The attack on PDF with perfect knowledge is chosen as an example to illustrate the complexity of the attack methods. Similar observations can be obtained from the rest of the experimental settings. Table 2 shows the average required times in millisecond (ms) on manipulating a sample in PDF to mislead ID3, FID3 and YST by using *our method*, *PPNT*, *RVTM*, *GAN1*, *GAN2*, Sub_{NN} and Sub_{SVM} with perfect knowledge. The results show that crafting attack samples by the generic attack methods is significantly more time consuming than the methods designed for DT. The time requirements of *our method* increases dramatically for FID3 and YST due to the complicated tree structures. On the other hand, since *GAN1*, *GAN2*, Sub_{NN} and Sub_{SVM} only consider the label output of the target DT, their running times are more stable.

Table 2
Comparison on the time of crafting an attack sample by using *our method*, *PPNT*, *RVTM*, *GAN1*, *GAN2*, Sub_{NN} and Sub_{SVM} in PDF (millisecond)

Classifier	Attack method						
	Our method	*PPNT*	*RVTM*	*GAN1*	*GAN2*	Sub_{NN}	Sub_{SVM}
ID3	1.68	0.30	0.11	326.01	222.41	29.05	217.6
FID3	533.74	–	–	325.86	222.37	121.17	243.82
YST	1135.05	–	–	326.20	222.25	124.53	203.51

5. Conclusion

This study provides the first investigation on how the fuzzifying process affects the robustness in an adversarial environment. The decision tree, which is a popular machine learning method, is used for demonstration. Gradient descent based evasion attack methods cannot be applied to DT as its decision function is non-differentiable. In order to evaluate the robustness of DT and FDT precisely, an attack model specially designed for DT is first proposed. The first evasion attack method designed for both DT and FDT is proposed to fill the gap of examining the security issue of fuzzy systems in adversarial environments. The proposed attack model is based on the sensitivity measure, which quantifies the influence of changing a feature on the output of DT.

The robustness of DT and FDT under evasion attack is investigated in the perfect and limited knowledge settings experimentally. Our proposed method is compared with *PPNT* and *RVTM*, which are state-of-the-art attack methods designed for DT, and the attack methods using surrogate and GAN models. The results indicate that our method downgrades ID3, FID3 and YST more effectively in the perfect knowledge setting when comparing them with other attack methods. When only partial knowledge on the target classifier is obtained, our method also achieves the largest influence on ID3 and YST. After showing the effectiveness of our attack method, the robustness of DT and FDT is then evaluated experimentally. The results suggest that the robustness is enhanced by the fuzzifying process, *i.e.*FDT is more robust than DT. Because the structure of a system becomes more complicated from the fuzzification process by considering more features in the decision making, more modification on a sample is required for a successful attack. We also found that FDT with more fuzzy membership functions is usually more vulnerable since considering more membership functions reduces the depth of a tree and less features are used. Therefore, less modification is required for a successful attack generally. These findings are consistent with the previous studies in adversarial learning.

One possible future work is to investigate the impact of different shapes of fuzzy membership functions on the robustness of FDT. Several common functions, *e.g.*Gaussian and trapezoidal, can be evaluated since the shape of a membership function is important in real applications. The fuzzifying process can also be evaluated by using a neural network, which is another commonly used classifier, in an adversarial environment. This may provide a countermeasure other than robust learning and data sanitization against adversarial attack.

CRediT authorship contribution statement

Patrick P.K. Chan: Conceptualization, Writing - review & editing, Funding acquisition. **Juan Zheng:** Writing - review & editing, Software, Validation. **Han Liu:** Methodology, Writing - original draft. **E.C.C. Tsang:** Investigation, Formal analysis. **Daniel S. Yeung:** Supervision, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by Natural Science Foundation of Guangdong Province, China (No. 2018A030313203), and the Fundamental Research Funds for the Central Universities, China (No. 2018ZD32).

References

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[2] A.L. Maas, A.Y. Hannun, A.Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in: Proc. Icml, Vol. 30, No. 1, 2013, p. 3.
[3] T.A. Meyer, B. Whateley, SpamBayes: Effective open-source, Bayesian based, email classification system, in: CEAS, Citeseer, 2004.
[4] M. Ståhlberg, Malware detection, uS Patent App. 12/462, 913, 2011.
[5] N. Dalvi, P. Domingos, S. Sanghai, D. Verma, et al., Adversarial classification, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2004, pp. 99–108.
[6] D. Lowd, C. Meek, Adversarial learning, in: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, ACM, 2005, pp. 641–647.
[7] L. Huang, A.D. Joseph, B. Nelson, B.I. Rubinstein, J. Tygar, Adversarial machine learning, in: Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence, ACM, 2011, pp. 43–58.
[8] B.A. Nelson, Behavior of Machine Learning Algorithms in Adversarial Environments, Tech. Rep., California Univ Berkeley Dept of Electrical Engineering and Computer Science, 2010.
[9] A. Kurakin, I. Goodfellow, S. Bengio, Adversarial examples in the physical world, 2016, arXiv preprint arXiv:1607.02533.
[10] L. Chen, Y. Ye, T. Bourlai, Adversarial machine learning in malware detection: Arms race between evasion attack and defense, in: Intelligence and Security Informatics Conference (EISIC), 2017 European, IEEE, 2017, pp. 99–106.
[11] K. Grosse, N. Papernot, P. Manoharan, M. Backes, P. McDaniel, Adversarial examples for malware detection, in: European Symposium on Research in Computer Security, Springer, 2017, pp. 62–79.
[12] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, F. Roli, Evasion attacks against machine learning at test time, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2013, pp. 387–402.
[13] B. Biggio, B. Nelson, P. Laskov, Poisoning attacks against support vector machines, 2012, arXiv preprint arXiv:1206.6389.
[14] B. Biggio, B. Nelson, P. Laskov, Support vector machines under adversarial label noise, in: Asian Conference on Machine Learning, 2011, pp. 97–112.
[15] J. Stallkamp, M. Schlipsing, J. Salmen, C. Igel, Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition, Neural Netw. 32 (2012) 323–332.
[16] B. Nelson, M. Barreno, F.J. Chi, A.D. Joseph, B.I. Rubinstein, U. Saini, C.A. Sutton, J.D. Tygar, K. Xia, Exploiting machine learning to subvert your spam filter, in: LEET, Vol. 8, 2008, pp. 1–9.
[17] G.F. Cretu, A. Stavrou, M.E. Locasto, S.J. Stolfo, A.D. Keromytis, Casting out demons: Sanitizing training data for anomaly sensors, in: Security and Privacy, 2008. IEEE Symposium on, SP 2008, IEEE, 2008, pp. 81–95.
[18] P.P. Chan, Z.-M. He, H. Li, C.-C. Hsu, Data sanitization against adversarial label contamination based on data complexity, Int. J. Mach. Learn. Cybern. 9 (6) (2018) 1039–1052.
[19] F. Zhang, P.P. Chan, B. Biggio, D.S. Yeung, F. Roli, Adversarial feature selection against evasion attacks, IEEE Trans. Cybern. 46 (3) (2016) 766–777.
[20] P.P. Chan, Z. Lin, X. Hu, E.C. Tsang, D.S. Yeung, Sensitivity based robust learning for stacked autoencoder against evasion attack, Neurocomputing 267 (2017) 572–580.
[21] A. Demontis, P. Russu, B. Biggio, G. Fumera, F. Roli, On security and sparsity of linear classifiers for adversarial settings, Lecture Notes in Comput. Sci. 10029 (2016) 322–332.
[22] B. Biggio, G. Fumera, F. Roli, Multiple classifier systems for robust classifier design in adversarial environments, Int. J. Mach. Learn. Cybern. 1 (1–4) (2010) 27–41.
[23] B. Biggio, G. Fumera, F. Roli, Evade hard multiple classifier systems, in: Applications of Supervised and Unsupervised Ensemble Methods, Springer, 2009, pp. 15–38.
[24] N. Papernot, P. McDaniel, I. Goodfellow, Transferability in machine learning: from phenomena to black-box attacks using adversarial samples, 2016, arXiv preprint arXiv:1605.07277.
[25] S.A. Mostafa, A. Mustapha, M.A. Mohammed, M.S. Ahmad, M.A. Mahmoud, A fuzzy logic control in adjustable autonomy of a multi-agent system for an automated elderly movement monitoring application, Int. J. Med. Inform. 112 (2018) 173–184.
[26] D.M. Magerman, Statistical decision-tree models for parsing, in: Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, 1995, pp. 276–283.
[27] M. Umanol, H. Okamoto, I. Hatono, H. Tamura, F. Kawachi, S. Umedzu, J. Kinoshita, Fuzzy decision trees by fuzzy ID3 algorithm and its application to diagnosis systems, in: Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the Third IEEE Conference on, IEEE, 1994, pp. 2113–2118.
[28] J.-H. Wang, P.S. Deng, Y.-S. Fan, L.-J. Jaw, Y.-C. Liu, Virus detection using data mining techinques, in: IEEE 37th Annual 2003 International Carnahan Conference OnSecurity Technology, 2003. Proceedings, IEEE, 2003, pp. 71–76.
[29] J.Z. Kolter, M.A. Maloof, Learning to detect malicious executables in the wild, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2004, pp. 470–478.
[30] K. Sachdeva, M. Hanmandlu, A. Kumar, Real life applications of fuzzy decision tree, Int. J. Comput. Appl. 42 (10) (2012) 24–28.
[31] X. Carreras, L. Marquez, Boosting trees for anti-spam email filtering, 2001, arXiv preprint cs/0109015.
[32] H. Chen, H. Zhang, S. Si, Y. Li, D. Boning, C.-J. Hsieh, Robustness verification of tree-based models, 2019, arXiv:1906.03849.
[33] P.P. Chan, C. Yang, D.S. Yeung, W.W. Ng, Spam filtering for short messages in adversarial environment, Neurocomputing 155 (2015) 167–176.

[34] B. Biggio, G. Fumera, F. Roli, Security evaluation of pattern classifiers under attack, IEEE Tran. Knowl. Data Eng. 26 (4) (2014) 984–996.
[35] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, F. Roli, Is feature selection secure against training data poisoning? in: International Conference on Machine Learning, 2015, pp. 1689–1698.
[36] G. Varghese, F.G. Bonomi, J.A. Fingerhut, Methods and systems to detect an evasion attack, uS Patent 8, 613, 088, 2013.
[37] A. Kurakin, D. Boneh, F. Tramèr, I. Goodfellow, N. Papernot, P. McDaniel, Ensemble adversarial training: Attacks and defenses, 2018.
[38] R. Alaifari, G.S. Alberti, T. Gauksson, ADef: an iterative algorithm to construct adversarial deformations, 2018, arXiv preprint arXiv:1804.07729.
[39] I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, in: International Conference on Learning Representations, 2015.
[40] S. Moosavidezfooli, A. Fawzi, O. Fawzi, P. Frossard, Universal adversarial perturbations, Comput. Vis. Pattern Recognit. (2017) 86–94.
[41] N. Papernot, P.D. Mcdaniel, I.J. Goodfellow, S. Jha, Z.B. Celik, A. Swami, Practical black-box attacks against machine learning, Comput. Commun. Secur. (2017) 506–519.
[42] N. Carlini, D. Wagner, Towards evaluating the robustness of neural networks, in: 2017 IEEE Symposium on Security and Privacy, SP, IEEE, 2017, pp. 39–57.
[43] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, A. Swami, The limitations of deep learning in adversarial settings, in: Security and Privacy (EuroS&P), 2016 IEEE European Symposium on, IEEE, 2016, pp. 372–387.
[44] B. Biggio, G. Fumera, F. Roli, Multiple classifier systems for adversarial classification tasks, in: International Workshop on Multiple Classifier Systems, Springer, 2009, pp. 132–141.
[45] F. Tramèr, F. Zhang, A. Juels, M.K. Reiter, T. Ristenpart, Stealing machine learning models via prediction APIs, in: USENIX Security Symposium, 2016, pp. 601–618.
[46] Y. Li, Y. Wang, Y. Wang, L. Ke, Y.A. Tan, A feature-vector generative adversarial network for evading PDF malware classifiers, Inform. Sci. 523 (2020).
[47] E. Alhajjar, P. Maxwell, N.D. Bastian, Adversarial machine learning in network intrusion detection systems, 2020, arXiv:2004.11898.
[48] C. Tu, P. Ting, P. Chen, S. Liu, H. Zhang, J. Yi, C. Hsieh, S. Cheng, AutoZOOM: Autoencoder-based zeroth order optimization method for attacking black-box neural networks, 2018, CoRR, abs/1805.11770.
[49] W. Brendel, J. Rauber, M. Bethge, Decision-based adversarial attacks: Reliable attacks against black-box machine learning models, 2017, ArXiv abs/1712.04248.
[50] J.R. Quinlan, Induction of decision trees, Mach. Learn. 1 (1) (1986) 81–106.
[51] Y. Yuan, M.J. Shaw, Induction of fuzzy decision trees, Fuzzy Sets Syst. 69 (2) (1995) 125–139.
[52] U. Fayyad, K. Irani, Multi-interval discretization of continuous-valued attributes for classification learning, 1993.
[53] K. Zou, W. Sun, H. Yu, F. Liu, ID3 decision tree in fraud detection application, in: Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on, Vol. 3, IEEE, 2012, pp. 399–402.
[54] C. Sinclair, L. Pierce, S. Matzner, An application of machine learning to network intrusion detection, in: Computer Security Applications Conference, 1999. (ACSAC'99) Proceedings. 15th Annual, IEEE, 1999, pp. 371–377.
[55] D. Yeung, X. Wang, E. Tsang, Learning weighted fuzzy rules from examples with mixed attributes by fuzzy decision trees, in: Systems, Man, and Cybernetics, 1999. IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on, Vol. 3, IEEE, 1999, pp. 349–354.
[56] B. Biggio, I. Corona, Z.-M. He, P.P. Chan, G. Giacinto, D.S. Yeung, F. Roli, One-and-a-half-class multiple classifier systems for secure learning against evasion attacks at test time, in: International Workshop on Multiple Classifier Systems, Springer, 2015, pp. 168–180.
[57] F. Sebastiani, Machine learning in automated text categorization, ACM Comput. Surv. 34 (1) (2002) 1–47.
[58] A. Kołcz, C.H. Teo, Feature weighting for improved classifier robustness, in: CEAS'09: Sixth Conference on Email and Anti-Spam, 2009.
[59] D. Dua, C. Graff, UCI Machine Learning Repository, University of California, Irvine, School of Information and Computer Sciences, 2017, URL http://archive.ics.uci.edu/ml.
[60] C.I. for Cybersecurity, NSL-KDD, Canadian Institute for Cybersecurity, 2009, URL https://www.unb.ca/cic/datasets/nsl.html.
[61] 2BiTT, Automatic training (decision tree) model, URL https://www.cnblogs.com/qwj-sysu/p/5935016.html.