Data Poisoning Attacks Against Outcome Interpretations of Predictive Models

Hengtong Zhang
Purdue University
University at Buffalo
htzhang.work@gmail.com

Jing Gao
Purdue University
jinggao@purdue.edu

Lu Su
Purdue University
lusu@purdue.edu

ABSTRACT

The past decades have witnessed significant progress towards improving the accuracy of predictions powered by complex machine learning models. Despite much success, the lack of model interpretability prevents the usage of these techniques in life-critical systems such as medical diagnosis and self-driving systems. Recently, the interpretability issue has received much attention, and one critical task is to explain *why a predictive model makes a specific decision*. We refer to this task as outcome interpretation. Many outcome interpretation methods have been developed to produce human-understandable interpretations by utilizing intermediate results of the machine learning models, such as gradients and model parameters.

Although the effectiveness of outcome interpretation approaches has been shown in a benign environment, their robustness against data poisoning attacks (i.e., attacks at the training phase) has not been studied. As the first work towards this direction, we aim to answer an important question: *Can training-phase adversarial samples manipulate the outcome interpretation of target samples?* To answer this question, we propose a data poisoning attack framework named IMF (**I**nterpretation **M**anipulation **F**ramework), which can manipulate the interpretations of target samples produced by representative outcome interpretation methods. Extensive evaluations verify the effectiveness and efficiency of the proposed attack strategies on two real-world datasets.

CCS CONCEPTS

• **Security and privacy → Software and application security**;
• **Human-centered computing → Visualization**.

KEYWORDS

Model Interpretation, Adversarial Learning

ACM Reference Format:
Hengtong Zhang, Jing Gao, and Lu Su. 2021. Data Poisoning Attacks Against Outcome Interpretations of Predictive Models. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3447548.3467405

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467405

1 INTRODUCTION

The past decade has witnessed the astonishing predictive power of machine learning models, especially with the development of complex models such as deep learning models. Their success has been demonstrated in a variety of real-world tasks, such as: image classification [12, 13, 37], object detection [18, 19] and machine translation [1, 21, 36]. Despite much success, the lack of model interpretability prevents the usage of these techniques in life-critical systems such as medical diagnosis and self-driving systems. Even when a model can learn complex patterns in data that enable highly accurate predictions, these predictions may not be useful without proper interpretations.

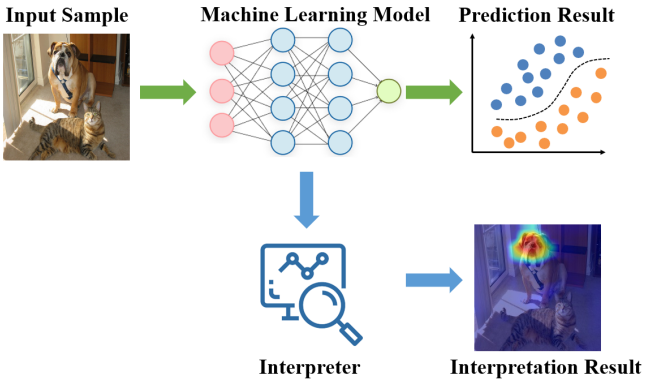


Figure 1: Workflow of an Outcome Interpretation Approach. The input sample (a dog) is fed into the machine learning model to get the prediction result. The interpreter utilizes the intermediate results produced by the machine learning model to interpret the decision. In this example, the interpreter highlights the dog's head. Such an interpretation result indicates that the dog's head is the most important region in the image.

Therefore, recently lots of efforts have been devoted to developing effective approaches that reveal the working mechanism of predictive models. Existing work can be roughly categorized into intrinsic model interpretation [5, 11, 17] and outcome interpretation [9, 20, 20, 25, 30, 44]. The former tries to provide explanations about the model structure that allows users to understand the logic of the model, while the latter reveals the reasons why a particular outcome prediction is made based on the model. In this paper, we focus on the outcome interpretation. The general workflow of outcome interpretation is demonstrated in Figure 1. Given a specific decision made by a predictive model, an outcome interpretation

approach seeks a human-understandable interpretation for the decision. Let us consider the case that an input sample (i.e., an image) is fed into a predictive model to receive the prediction of its class (i.e., dog). The interpreter utilizes the intermediate results produced by the predictive model to estimate the importance degrees of the features (raw features or intermediate features) on the final prediction result. In this example, the most important features are the pixels that form the dog's head in the image. This interpretation result indicates that the dog's head is the most important region that leads to the predicted class (i.e., dog) according to the predictive model.

Although these approaches can generate meaningful outcome interpretations, they also open up new possibilities for adversaries to conduct attacks. Attacks towards the interpretation results are feasible because they rely on the intermediate results from its corresponding predictive model, which can be manipulated by injecting carefully crafted poisoning samples into the training set. As a result, the interpretation result of a targeted test sample may be reshaped. Needless to say, such attacks can lead to unaffordable consequences as a wrong interpretation result may mislead end users' decisions. Especially when the victim model and the interpretation approaches are adopted in life-critical applications, such as medical diagnoses, such attacks may result in severe damages. Consider the scenario that a predictive model is trained to diagnose the disease based on medical images. The model provides medical specialists its prediction, and the interpreter generates an interpretation of the result, e.g., the location of the lesion, to support the diagnosis results. When the attack is conducted, the location of the highlighted region can be shifted, and a normal region may be marked as a critical disease-related area. As a result, the medical specialist may trust the *manipulated interpretation* result and further conduct incorrect treatments.

Unfortunately, to the best of our knowledge, there is very little research effort investigating the vulnerabilities of outcome interpreters facing data poisoning attacks. We acknowledge that there are some existing works [35, 39, 42] studying the vulnerability of outcome interpretation approaches. Still, all of them focused on strategies for evasion attacks, which are conducted during the testing phase. In contrast, data poisoning attacks are enabled in the training phase by modifying the training set. As they occur at different phases, the attack mechanisms of evasion and data poisoning attacks are quite different. Typically, the attack approaches designed for evasion settings cannot be directly used to launch data poisoning attacks.

In this paper, we perform the first systematic investigation of the potential vulnerability of existing outcome interpretation methods facing sophisticated data poisoning attacks. The proposed attack framework IMF (**I**nterpretation **M**anipulation **F**ramework) poisons the training set of the target model by perturbing and injecting some specific samples to manipulate the interpretation results of targeted samples. The proposed framework formulates the poisoning attack as a bi-level optimization problem, which intends to use the crafted poisoning samples to encircle the target sample in the feature space and "drag" the interpretation of the target sample towards the result that the attacker desires. The upper-level problem evaluates the soundness of the current crafted poisoning sample, while the lower-level problem updates model parameters given these poisoning samples. To solve this bi-level optimization problem, we propose an efficient iterative learning algorithm that alternates between poisoning sample generation and model parameter estimation in each iteration. We evaluate the effectiveness of the proposed IMF in two real-world applications against two different types of outcome interpreters. Experimental results demonstrate that the poisoning attack method proposed in this paper can successfully reshape the interpretations of target samples, which are generated by existing outcome interpreters.

The contributions of this paper are summarized as follows:

- We identify the potential vulnerability of existing outcome interpretation methods facing sophisticated data poisoning attacks.
- We design a general poisoning attack framework *IMF* to generate highly effective poisoning samples against existing outcome interpretation approaches efficiently.
- We conduct extensive experiments against two representative interpretation approaches in two real-world applications to demonstrate the advantages of *IMF*.

The rest of the paper is organized as follows. Section 2 provides a review of related work. In Section 3, we introduce the background knowledge and detail the threat model. Then we describe the proposed framework *IMF* in Section 4. In Section 5, we conduct a series of experiments and case studies on real-world datasets. We conclude the paper in Section 6.

2 RELATED WORK

In this section, we survey the related work from three aspects: outcome interpretation methods, the data poisoning attack methods against machine learning models, and the adversarial attacks against interpretation methods.

Outcome Interpretation Methods: Outcome interpretation methods aim to identify the contribution of each feature in the inputs and further explain why the machine learning models make a specific decision. Outcome interpretation methods can be classified as model-agnostic methods and model-specific methods. Model-agnostic methods treat the targeted machine learning models as black-boxes and generate interpretations without accessing the intermediate representations or model parameters. The most representative series of model-agnostic interpretation approaches are based on local approximation [25, 26]. These approaches typically learn simple and interpretable white-box models (e.g., linear models or decision trees) to approximate the targeted machine learning model. The importance of each feature is then obtained from the parameters of the interpretable white-box model.

Apart from model-agnostic interpretation methods, there are also interpretation methods that closely couple with specific machine learning models. These methods treat machine learning models as white-boxes and utilize information like intermediate layers' outputs or learned model parameters to derive interpretations. For instance, back-propagation-guided interpretation approaches [30, 33, 34] use the gradient of model output with respect to the input features to estimate the influence of each input feature on generating the current prediction. Representation-guided interpretations [28, 44] utilize the high-level feature map and the final fully connect layer to compute the importance of different regions in the feature map given specific classification predictions.

Mask-based approaches [7, 9] generate interpretations via mask perturbations and gradient descent optimization. These methods learn a perturbation mask over input features via solving an optimization problem. In this paper, we focus on analyzing the vulnerability of *back-propagation-guided approaches* and *representation-guided approaches* and leave vulnerability assessment of the other approaches as our future work.

Data Poisoning Attacks against Machine Learning Models: Data poisoning attacks against machine learning models have become an important research topic in the field of adversarial machine learning. This type of attack takes place during the training stage of machine learning models. The attacker tries to contaminate the training data by injecting well-designed samples to force a nefarious model on the learner. Data poisoning attacks [2, 4, 10, 16, 23, 24, 29, 40, 41] have been studied against a wide range of learning systems. The first work in this field [2] investigates the vulnerability of support vector machine (SVM), where an attacker progressively injects malicious data points to the training set to maximize the classification error. Later, Mei et al. [23] generalize these attacks against SVM into a bi-level optimization framework against general offline learners with a convex objective function. Recently, [16, 24] propose poisoning attack strategies for neural networks. Chen et al. [4] propose targeted backdoor attacks, which cause the classifier to fail on test examples with specific patterns. Gu et al. [10] train a network using mislabeled images tagged with a special pattern, causing the classifier to learn the association between the pattern and the class label. All the existing works has been focused on manipulating the classification results of specific target samples. Unlike these works, we focus on a more challenging task, i.e., *manipulating the interpretations of specific target samples.*

Adversarial Attack against Model Interpretation: Attacking the model interpretation as a new topic is not intensively studied. To the best of our knowledge, there are only limited works [35, 39, 42] investigating this specific problem. Zhang et al. [42] propose an attack approach named ADV^2, which is built upon the classic PGD [22] framework, to craft adversarial samples that can conduct evasion attacks against four outcome interpretation methods. Subramanya et al. [35] propose an optimization problem to craft the adversarial patch and paste the patch on the clean image. The patch suppresses Grad-CAM activation at the location of the patch. Warnecke et al. [39] first introduce criteria to evaluate interpretation methods designed for computer security applications and then use these criteria to assess their robustness of these interpretation methods.

Witness of these adversarial threats, recent literature proposes multiple approaches [3, 27, 31] to enhance the robustness of interpretation methods. For instance, [31] tackles the robustness issues by maximizing the alignment between the input image and its saliency map using soft-margin triplet loss. [3] discovers that saliency explanations provided by Bayesian Neural Networks are more stable under adversarial contexts. [27] obtains robust interpretations by simply aggregating results from multiple interpretation methods. The major difference between these pieces of literature and this paper is that these methods focus on the context of test-phase (evasion) attacks, but the method proposed in this paper is designed for training-phase (data poisoning) attacks. To the best of our knowledge, there is only one parallel work [8] discussing the robustness of interpretation methods when facing training-phase attacks. [8] proposes to add backdoor triggers into some specific categories of training samples to manipulate the interpret results of the corresponding categories, while our work injects extra training samples to precisely control the interpretation of some particular test samples. Our problem settings and methodologies are fundamentally different.

3 BACKGROUND & THREAT MODEL

In this section, we first formulate the outcome interpretation task and briefly introduce several representative approaches. After that, we describe the threat model, which specifies the attack goal, attack approach, and attacker's capability.

3.1 Background

To facilitate the discussions, we would like to introduce some concepts that are used in the rest of this paper.

The outcome interpretation task involves an input sample $\boldsymbol{X}$, a machine learning model $f(\boldsymbol{X};\boldsymbol{\Theta})$ parameterized by $\boldsymbol{\Theta}$ and an outcome interpreter $G(\boldsymbol{X}; f; \boldsymbol{\Theta})$, which is coupled with f and also use the parameters of $f(\boldsymbol{X};\boldsymbol{\Theta})$. In this paper, we focus on the machine learning models for classification tasks, in which the classifier takes the sample $\boldsymbol{X}$ as input and outputs a class c. The probability of assigning sample $\boldsymbol{X}$ to class c is denoted as $f_c(\boldsymbol{X};\boldsymbol{\Theta})$. With these notations and concepts, we formulate the outcome interpretation task as follows:

Definition 3.1 (Outcome Interpretation Task). Outcome interpreter provides a human-understandable interpretation of the classifier's prediction for a specific sample. In this paper, we assume such interpretations are given in the form of saliency maps. The interpreter G generates an attribution map $\boldsymbol{m} = G(\boldsymbol{X}; f; \boldsymbol{\Theta})$, with its i-th element $\boldsymbol{m}[i]$ quantifying the importance of $\boldsymbol{X}$'s i-th feature with respect to the model prediction $f(\boldsymbol{X};\boldsymbol{\Theta})$. Some interpreters can also output the interpretation with respect to a specific class c, we denote these interpreters as $G_c(\boldsymbol{X}; f; \boldsymbol{\Theta})$.

In this paper, we consider two major types of outcome interpreters: *back-propagation-guided interpreters* and *representation-guided interpreters.*

Back-Propagation-Guided Interpreters. Back-propagation-guided interpreters utilize the gradient (or its variants) of the model prediction with respect to the input feature of a given sample to evaluate the importance of each input feature. The intuition behind this category of methods is that a larger gradient magnitude indicates a higher relevance of the feature to the prediction. In this section, we consider gradient saliency (GRAD) [30] as the representative methods of this category. GRAD utilizes the model prediction (probability) $f_c(\boldsymbol{X})$ for a given input $\boldsymbol{X}$ and a given class c to derive the interpretation map, i.e.:

$$G_c^{GRAD}(\boldsymbol{X}; f; \boldsymbol{\Theta}) = \left| \frac{\partial f_c(\boldsymbol{X})}{\partial \boldsymbol{X}} \right|. \tag{1}$$

Representation-Guided Interpreters. Representation-guided interpreters leverage the feature maps at intermediate layers of DNNs to generate attribution maps. We consider class activation mapping (CAM) [44] as a representative interpreter of this class.

For image data, we use $a_k[i,j]$ to denote the activation value at the spatial position (i,j) of channel k in the last convolutional layer. The global average pooling of the k-th channel is defined as:

$$A_k = \sum_{i=1}^{m}\sum_{j=1}^{n} a_k[i,j], \tag{2}$$

where m and n denote the width and length of A_k. Let $w_{k,c}$ be the parameter of the connection between the k-th input and the c-th output of the final linear layer, CAM is defined as:

$$G_c^{CAM}(\boldsymbol{X}; f; \boldsymbol{\Theta})[i,j] = \sum_k w_{k,c} a_k[i,j], \tag{3}$$

where $G_c^{CAM}(\boldsymbol{X}; f; \boldsymbol{\Theta})[i,j]$ denotes the CAM value at position $[i,j]$. For general multivariate data, we just replace the 2-D activation matrix with a 1-D vector, which also denotes the activation value of features in the final feature layer.

3.2 Threat Model

We have briefly described the representative approaches for the outcome interpretation task. Now let us detail the threat model of the poisoning attack against these approaches.

Attack Goal: The goal of the attacker is to manipulate the specific targeted samples' interpretation results but at the same time keep the classification performance.

Attack Approach: To achieve the attack goal, we consider the most general scenario in which the attackers can add poisoning samples (including modifications and injections) into the training set of a specific machine learning system.

Knowledge of Attackers: Here, we follow the white-box setting, in which the attacker has the knowledge of the architecture and parameters of both the targeted interpreter and the corresponding machine learning model. These are common settings of existing work investigating data poisoning attacks. We also assume the number of training samples the attacker can poison is small. Given the general threat model, the problem we investigate in this paper can be formally defined as follows:

Definition 3.2 (Problem Definition). Consider a machine learning model $f(\boldsymbol{X}; \boldsymbol{\Theta})$ parameterized by $\boldsymbol{\Theta}$ for the classification task, which takes a sample $\boldsymbol{X}$ as input and outputs the extracted feature. The extracted feature is followed by a softmax layer to produce a logit label y_x. The goal of interpretation poisoning is to craft training-stage poisoning samples to manipulate the outcome interpretation of the targeted sample $\boldsymbol{T}$, without much degrading the classification performance. In the rest of this paper, we use $\boldsymbol{G}$ to denote the attacker's desired interpretation result[1] for the targeted sample $\boldsymbol{T}$.

4 POISONING ATTACK

In this section, we describe the proposed **I**nterpretation **M**anipulation **F**ramework (IMF) for crafting training-stage poisoning samples to manipulate the outcome interpretations of targeted test samples. We formulate the interpretation manipulation problem as a bi-level optimization problem. Under this general framework, we introduce effective and efficient solutions, and discuss important issues to make the framework practical.

[1]Note: the desired interpretation may only involve part of the features instead of all the features.

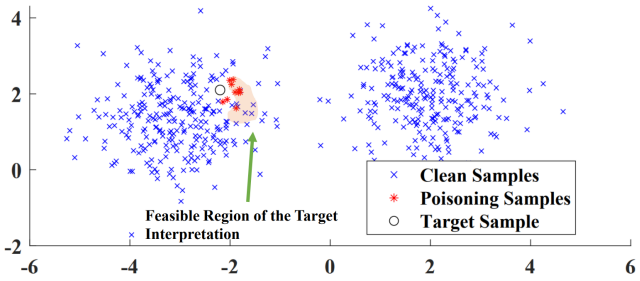


Figure 2: An Illustration of the Clean Samples, Poisoning Samples and the Targeted Sample.

4.1 IMF for Crafting Poisoning Samples

The IMF proposed in this paper follows the *feature "encirclement"* strategy, which exploits the imperfections in the feature extraction layers of the deep models. In brief, we propose to inject multiple poisoning samples into the training set of the classifier to encircle the targeted test sample in the feature space closely. Thus, the injected poisoning samples will have large influences on the targeted sample and indirectly force the targeted sample to have desired interpretations. An overview of such a poisoning attack is illustrated in Figure 2.

Concretely, we first pick base samples which are typically (1) samples with the same class label as the targeted test sample $\boldsymbol{T}$ in the training set; and (2) the targeted test sample itself. Then we conduct perturbations on these base samples to craft poisoning samples. We choose to craft poisoning samples via slightly modifying base samples because it is challenging to generate realistic samples like images from scratch. Formally, we define the following optimization objective to craft the poisoning sample:

$$\min_{\boldsymbol{P}} \ ||G_c(\boldsymbol{T}; f; \boldsymbol{\Theta}) - \boldsymbol{G}||^2 + \lambda_1 ||\boldsymbol{P} - \boldsymbol{T}||^2 + \lambda_2 L(\boldsymbol{T}, c, \boldsymbol{\Theta}), \tag{4}$$

where $\boldsymbol{T}$ is the targeted test sample. $\boldsymbol{G}$ denotes the desired interpretation of $\boldsymbol{T}$ specified by the attacker. $\boldsymbol{P}$ is the poisoning sample, c is the classification label of $\boldsymbol{T}$. $G_c(\boldsymbol{T}; f; \boldsymbol{\Theta})$ denotes the targeted interpretation which may utilize the intermediate results of f and therefore can also be treated as parameterized by $\boldsymbol{\Theta}$. L is the loss function of the classifier. λ_1, λ_2 are coefficients. The ideas behind each term within Eq. (4) are as follows. The *first term* forces the interpretation of the targeted sample to be close to that specified by the attacker, and the *second term* forces the poisoning sample to be close to the targeted sample to enlarge its influence on the targeted sample. The *third term* lets the label of the targeted sample remain unchanged so that the poisoning samples do not ruin the classification result.

However, there is a major limitation in Eq. (4). The poisoning samples are obtained based on the current classifier $f(\boldsymbol{P}; \boldsymbol{\Theta})$, which however, needs to be retrained on the contaminated data. After retraining, the model parameters of $\boldsymbol{\Theta}$ will also be changed. With the model parameters being changed, the poisoning samples crafted via Eq. (4) may no longer be able to achieve the desired interpretation $\boldsymbol{G}$. To tackle this issue, we propose to jointly optimize model parameters and craft poisoning samples via solving the following

bi-level optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{P}} \ ||G_c(\boldsymbol{T}; f; \hat{\boldsymbol{\Theta}}) - \boldsymbol{G}||^2 + \lambda_1||\boldsymbol{P} - \boldsymbol{T}||^2 + \lambda_2 L(\boldsymbol{T}, c, \hat{\boldsymbol{\Theta}}) \\ &s.t. \ \hat{\boldsymbol{\Theta}} = \arg\min_{\boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}), \end{aligned} \tag{5}$$

where $\mathcal{L}$ denotes the loss function parameterized by $\boldsymbol{\Theta}$. $\boldsymbol{X}_c$, $\boldsymbol{X}_p$, $\boldsymbol{Y}_c$, $\boldsymbol{Y}_p$, stand for the set of clean training samples, poisoning samples, and their labels, respectively. For simplicity, in the rest of this paper, we denote the upper-level and the lower-level objective function as $\mathcal{L}_{adv}$ and $\mathcal{L}_{train}$, respectively. Eq. (5) merely show the case of crafting a single poisoning sample, but it is straightforward to it generalize to a craft a set of poisoning samples.

Eq. (5) jointly considers the poisoning sample crafting as well as the model parameters optimization. Nevertheless, bi-level optimization problems like Eq. (5) are generally hard to solve, especially when both the upper-level and lower-level problems involve complex non-convex terms introduced by deep neural networks. In the rest of this section, we present efficient approximate general solutions of Eq. (5).

4.2 Learning of IMF

As one can see, the upper-level and the lower-level optimization problem are tightly coupled through the crafted poisoning samples, and the estimated model parameters $\boldsymbol{\Theta}$. Given the fact that the lower-level problem usually involves a complex deep neural network classifier, evaluating the exact gradient of $\boldsymbol{X}_p$ and $\boldsymbol{\Theta}$ can be prohibitively costly. To make things worse, since the scale and the optimal regions of the upper-level and lower-level problem are different, alternative training these two problems directly may cause severe fluctuations on $\boldsymbol{X}_p$ and $\boldsymbol{\Theta}$. Motivated by [14, 32, 38, 43], we resort to iterative learning algorithm to solve Eq. (5). *The learning algorithm consists of three steps in each poisoning sample craft iteration.*

Step 1: *First, we derive the ideal parameter update that can decrease the adversarial objective function $\mathcal{L}_{adv}$ (i.e., the upper-level of Eq.* (5)*).* Such an update will push the learned model towards the attack goal. Mathematically, the fastest direction of decreasing $\mathcal{L}_{adv}$ is opposite to its partial derivative w.r.t the model parameters. Thus, considering the adversarial objective, the update of model parameters can be estimated as:

$$\boldsymbol{\Theta}^{(r*)} \leftarrow \boldsymbol{\Theta}^{(r-1)} - \alpha \nabla_{\boldsymbol{\Theta}} \mathcal{L}_{\text{adv}}(\boldsymbol{X}_c^{(r)} \cup \boldsymbol{X}_p^{(r)}, \boldsymbol{Y}_c^{(r)} \cup \boldsymbol{Y}_p^{(r)}; \boldsymbol{\Theta}^{(r-1)}), \tag{6}$$

where $\boldsymbol{\Theta}^{(r)}$ is the parameter in the r-th iteration.

Step 2: *In the second step, we craft the poisoning samples that can manipulate the model parameters towards the desired direction, i.e., Eq.* (6). From the view of optimization, if we want the learning algorithm to update the model parameter from $\Theta^{(r-1)}$ to $\Theta^{(r*)}$, the following condition should be satisfied:

$$\begin{aligned} &\mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p + \boldsymbol{\delta}_X, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r*)}) \\ &< \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p + \boldsymbol{\delta}_X, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r-1)}), \end{aligned} \tag{7}$$

where $\boldsymbol{X}_c$, $\boldsymbol{X}_p$, $\boldsymbol{Y}_c$, and $\boldsymbol{Y}_p$ are mini-batches of samples. $\boldsymbol{\delta}_X$ is the update of $\boldsymbol{X}_p$ of the current step. The intuition behind condition Eq. (7) is straightforward: the poisoning samples should favor the updated parameters (i.e., $\boldsymbol{\Theta}^{(r*)}$) over the original ones (i.e., $\boldsymbol{\Theta}^{(r-1)}$). Otherwise, the parameters will not be shifted.

Based on condition Eq. (7), we approximate its both sides via first-order Taylor expansion at X_p:

$$\begin{aligned} &\mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r*)}) \\ &\quad + \boldsymbol{\delta}_X \frac{\partial \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r*)})}{\partial \boldsymbol{X}_p} \\ &< \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r-1)}) \\ &\quad + \boldsymbol{\delta}_X \frac{\partial \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r-1)})}{\partial \boldsymbol{X}_p}, \end{aligned} \tag{8}$$

which can be further reorganized as:

$$\begin{aligned} &\boldsymbol{\delta}_X \Bigg(\frac{\partial \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r*)})}{\partial \boldsymbol{X}_p} \\ &\quad - \frac{\partial \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r-1)})}{\partial \boldsymbol{X}_p} \Bigg) \\ &< \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r-1)}) \\ &\quad - \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r*)}). \end{aligned} \tag{9}$$

Obviously, the less the left-hand side of Eq. (9), the larger chance that condition Eq. (7) satisfies. Thus, the optimal $\boldsymbol{X}_p^{(r)}$ should move from $\boldsymbol{X}_p^{(r-1)}$ along the direction:

$$\begin{aligned} \boldsymbol{\delta}_X = &- \frac{\partial \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r*)})}{\partial \boldsymbol{X}_p} \\ &+ \frac{\partial \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}^{(r-1)})}{\partial \boldsymbol{X}_p}, \end{aligned} \tag{10}$$

and we can update $\boldsymbol{X}_p$ as:

$$\boldsymbol{X}_p^{(r)} \leftarrow \boldsymbol{X}_p^{(r-1)} + \gamma \boldsymbol{\delta}_X. \tag{11}$$

Step 3: In the final step, we update the model parameter for a single step using the lower-level objective function and the current mini-batch.

Through these three steps, we can alternatively update the poisoning samples and the model parameters until they converge.

4.3 Ensemble Mechanism for Robust Poisoning Samples Crafting

When victims train their deep models on the contaminated dataset, the randomnesses in the training process, such as parameter initialization and mini-batch orders, may hurt the effectiveness of the poisoning samples. Hence, using a single network for poisoning samples crafting may cause the crafted samples to over-fit to a specific optimization trajectory and hurt the "generalization ability" of these samples. Motivated by [14, 43], we revise the three-step learning algorithm described in Section 4.2 by ensembling the intermediate optimization results of a collection of differently initialized models, to alleviate the issues mentioned above.

Particularly, in Step 1, we introduce K differently initialized models, with the k-th model parameterized as Θ_k. Thus, Eq. (6) becomes:

$$\boldsymbol{\Theta}_k^{(r*)} \leftarrow \boldsymbol{\Theta}_k^{(r-1)} - \alpha \nabla_{\boldsymbol{\Theta}_k} \mathcal{L}_{\text{adv}}(\boldsymbol{X}_c^{(r)} \cup \boldsymbol{X}_p^{(r)}, \boldsymbol{Y}_c^{(r)} \cup \boldsymbol{Y}_p^{(r)}; \boldsymbol{\Theta}_k^{(r-1)}). \tag{12}$$

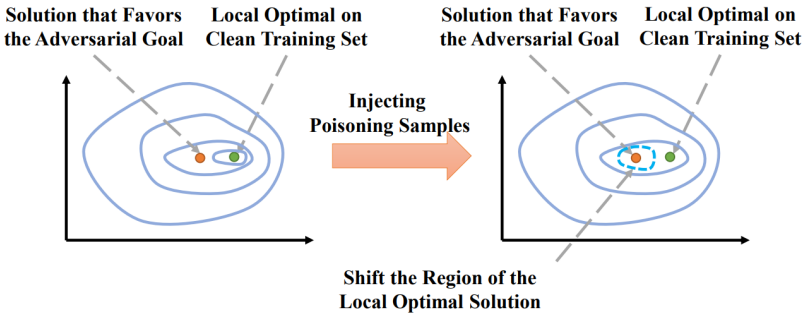


Figure 3: An Illustration of the Mechanism that Poisoning Samples Shift the Model Parameters.

Then in Step 2, we calculate $\boldsymbol{\delta}_X$ by averaging the optimal directions derived using the parameters of K differently initialized models, i.e.,

$$\boldsymbol{\delta}_X = -\frac{1}{K}\sum_{k=1}^{K}\left(\frac{\partial \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}_k^{(r*)})}{\partial \boldsymbol{X}_p} - \frac{\partial \mathcal{L}_{\text{train}}(\boldsymbol{X}_c \cup \boldsymbol{X}_p, \boldsymbol{Y}_c \cup \boldsymbol{Y}_p; \boldsymbol{\Theta}_k^{(r-1)})}{\partial \boldsymbol{X}_p}\right). \tag{13}$$

Such a direction ensemble method can significantly improve the generalization ability of poisoning samples empirically. Finally, in Step 3, we update the parameters of each model for a single step using the current mini-batch via:

$$\boldsymbol{\Theta}_k^{(r)} \leftarrow \boldsymbol{\Theta}_k^{(r-1)} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{train}}(\boldsymbol{X}_c^{(r)} \cup \boldsymbol{X}_p^{(r)}, \boldsymbol{Y}_c^{(r)} \cup \boldsymbol{Y}_p^{(r)}; \boldsymbol{\Theta}_k^{(r-1)}). \tag{14}$$

The learning algorithm with model ensembles is outlined in Algorithm 1.

4.4 Summary

In this section, we formulate a bi-level optimization problem Eq. (5) to craft poisoning samples for interpretation manipulation. The optimization objective perturbs base samples so that the perturbed samples locate around the targeted sample in feature space and makes the interpreter produce an interpretation close to the attacker's expectation. In this way, the targeted sample is encircled in the feature space, and its interpretation is reshaped as desired. To solve the bi-level problem, we propose an algorithm that iteratively updates the model parameters and crafts new poisoning samples. In the algorithm, we adopt the lookforward strategy to reduce the value fluctuation of parameters and poisoning samples among iterations. We also propose to ensemble multiple differently initialized models to improve the robustness of poisoning sample crafting.

Algorithm 1: Algorithm of Solving Eq. (5) and Crafting Poisoning Samples

Input : Training set $(\boldsymbol{X}, \boldsymbol{Y})$ of size N, targeted sample $\boldsymbol{T}$, poisoning budget $n \ll N$, number of crafting epochs E.
Output : Crafted poisoning sample $\boldsymbol{X}_p$.

Pre-train the models on the clean training set;
Initialize the poisoning samples. We denote the set of clean samples and poisoning samples as $\boldsymbol{X}_c$ and $\boldsymbol{X}_p$, respectively;
for *epoch* $= 1, \cdots, E$ **do**
  for $r = 1, \cdots, R$ *crafting iterations* **do**
    for *each* $k = 1, \cdots, K$ *models* **do**
      Cache current model parameter $\boldsymbol{\Theta}^{(r-1)}$;
      Update model parameter with the r-th batch via Eq. (12) ;
    end
    Use the model parameter $\boldsymbol{\Theta}^{(r)}$ to craft the poisoning sample $\boldsymbol{X}_p^{(r)}$ via Eq. (11) ;
    Project poisoning sample $\boldsymbol{X}_p^{(r)}$ to feasible region: $\boldsymbol{X}_p^{(r)} \leftarrow Proj(\boldsymbol{X}_p^{(r)}, \epsilon)$;
    for *each* $k = 1, \cdots, K$ *models* **do**
      Restore parameter $\boldsymbol{\Theta}^{(r-1)}$ for the k-th model ;
      Update the parameters of the k-th model with the current (i.e., r-th) mini-batch via Eq. (14) ;
    end
  end
end
return $\boldsymbol{X}_p$;

5 EXPERIMENTS

In this section, we conduct a series of experiments to evaluate the effectiveness of the proposed attack approach. We first describe the dataset and experiment setups in Section 5.1 and 5.2, respectively. Then we present experiment results and case studies in Section 5.3.

5.1 Datasets

We evaluate the effectiveness of the proposed IMF on two real-world datasets. The task on both datasets are classification.

- **Stanford Dogs Dataset** [15]. The Stanford Dogs dataset contains images of 120 breeds of dogs from around the world. The dataset consists of 20,580 images categorized into 120 classes. All the images are center-cropped to 224×224 pixels.
- **Osteoporotic Fracture Dataset** [6] This is the dataset released by UCSF for osteopathic study. This dataset consists of medical records on 9,703 women aged 65 years and older. These medical records are in the form of multivariate numerical features.

In this section, we randomly divide these datasets into training, validation, and test sets using 90/5/5 split. The poisoning samples crafted by the attack approach are injected into the training sets to manipulate the interpretations of certain targeted samples, which are randomly selected from the test set. In this paper, the number of targeted samples for each dataset is 10. *It is worth emphasizing that the test set in this paper is merely used as a pool to pick targeted samples and has nothing to do with the attack outcome evaluation.*

5.2 Experimental Settings

5.2.1 Classifiers. The datasets used in this paper consist of an image dataset, i.e., *Stanford Dogs*, and a multivariate numerical dataset, i.e., *Osteoporotic Fracture*. We adopt different classifiers for these two types of datasets. For the image dataset, we the representative ResNet-13 [12] as the classifier. Instead of training from scratch on the datasets used in this paper, we adopt the model weights pre-trained on the ImageNet dataset and finetune for our task. This finetune approach is commonly used in the computer vision domain. For the multivariate numerical dataset, we adopt a two-layer DNN, which is composed of two fully connected layers, as the classifier.

5.2.2 Targeted Interpretation Methods. We adopt GRAD [30] and CAM [44], the representatives of back-propagation based and representation based interpretation methods. The details of these methods are already mentioned in Section 3. We adopt their open-source implementation from the original authors. The parameters of targeted methods are set according to their papers' suggestions.

5.2.3 Attack Settings. Our implementation is based on Pytorch. For the proposed attack framework, the step size α is set to 0.001 and the parameter ϵ in Algorithm 1 is set to $\epsilon = 0.1$. The number of poisoning samples required for different targeted samples may vary, but typically below 10% of the total training samples. In practice, such insignificant perturbation thresholds will not make poisoning samples look artificial for both datasets.

5.2.4 Comparison Method. As there are no existing studies on the vulnerability of model interpretations against data poisoning attack, we compare the proposed IMF framework mechanism with WM (watermarking) attack. In the WM attack, we craft the poisoning image by taking a weighted combination between (1) the targeted test sample, and (2) the Top-10 training samples $\boldsymbol{g}_{sub}$ whose interpretations are closest to the attacker's desired interpretation.

5.3 Attack Effectiveness

According to the threat model defined in Section 3, the poisoning attack proposed in this paper aims at manipulating the interpretation of the targeted sample with the classification result of the sample unchanged.

5.3.1 Quantitative Evaluations. Since the interpreters discussed in this paper focus on quantifying the importance of features in the targeted sample and the importance scores are continuous values, we consider to use two metrics to measure the effectiveness of interpretation manipulations. [2] Specifically, given a targeted sample $\boldsymbol{t}$, we first propose to use averaged L_1 *distance* between the manipulated interpretation and the attacker's desired interpretation for evaluation. We normalize all the interpretations to $[0, 1]$ by dividing them by the number of dimensions. The smaller L_1 distance, the better the attack effectiveness is.

Moreover, apart from the L_1 distance, the manipulated interpretation and the attacker's desired interpretation should highlight a similar subset of features. Therefore, we propose to binarize the manipulated interpretation and the attacker's desired interpretation, and use the Jaccard similarity between them as the evaluation metric. Formally, we define $Norm(x)$ as a normalization function which shifts the scale of x into $[0, 1]$. Then the Jaccard similarity between the manipulated interpretation and the attacker's desired interpretation is given as:

$$Jaccard := \frac{|Norm(G_{adv}(\boldsymbol{t})) \cap Norm(G_{tgt}(\boldsymbol{t}))|}{|Norm(G_{adv}(\boldsymbol{t})) \cup Norm(G_{tgt}(\boldsymbol{t}))|}.$$

The larger the Jaccard similarity, the better attack effectiveness is.

Table 1: L_1 score and *Jaccard* score of the manipulated interpretation with respect to the attacker's desired interpretations on Stanford Dogs dataset.

	L1		Jaccard	
	GRAD	CAM	GRAD	CAM
WM	0.79	0.71	0.05	0.07
IMF	0.57	0.42	0.22	0.31

In Table 1 and Table 2, we report the L_1 distance and *Jaccard* similarity of the manipulated interpretation with respect to the attacker's desired interpretations on both datasets. As can be seen, our proposed IMF gets more than 20% lower L_1 scores than the baseline method for all the interpreters. This phenomenon indicates that the poisoning samples crafted by IMF can make the interpretations of the targeted samples closer to the attacker's desired interpretations. Moreover, we also observe that the *Jaccard* scores of the proposed IMF is also higher than those of the baseline. This means that the interpretations manipulated by IMF highlight a more similar subset of features than the interpretations manipulated by WM. From our view, the reason that WM suffers worse performance is that WM is designed for classification poisoning attacks and cannot be used to manipulate the interpretations.

Table 2: L_1 score and *Jaccard* score of the manipulated interpretation with respect to the attacker's desired interpretations on Osteoporotic Fracture dataset.

	L1		Jaccard	
	GRAD	CAM	GRAD	CAM
WM	0.87	0.79	0.03	0.17
IMF	0.65	0.50	0.08	0.40

5.3.2 Visualization Evaluation. Apart from the quantitative results above, we also visualize (1) the normal interpretations and (2) the manipulated interpretations of a test image from the Stanford Dogs dataset in Figure 4 as a case study. In Figure 4, for both GRAD and

[2]Note: We discretize the results of GRAD to make the results comparable with CAM.

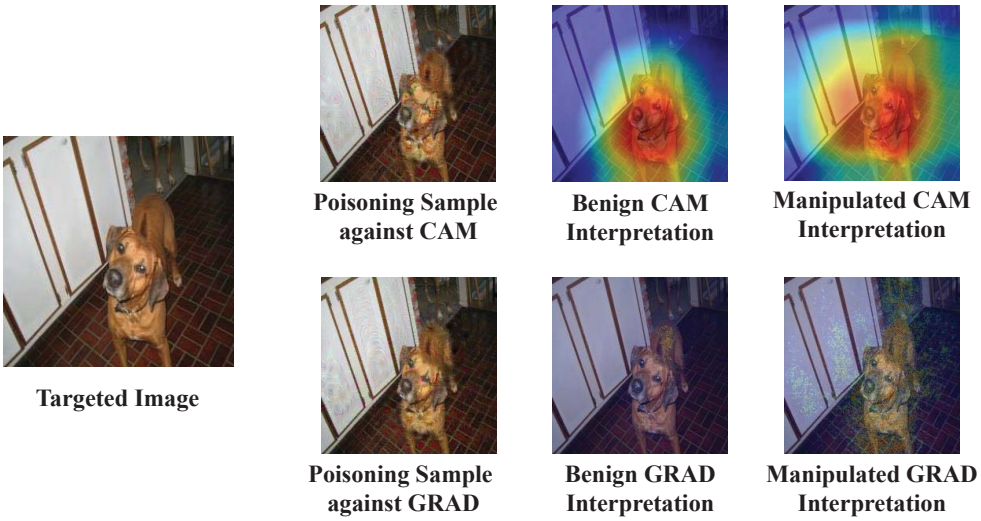
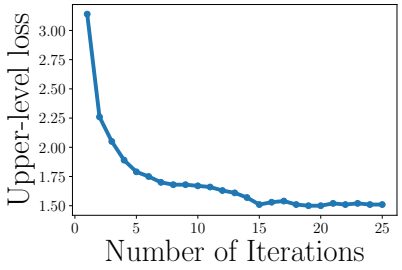


Figure 4: An example of the manipulated interpretations on Stanford Dogs Dataset. (Better be viewed with color).

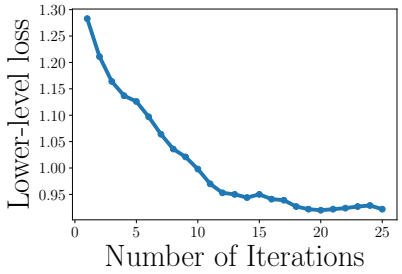
CAM interpreters, the important regions that lead to the classification results are highlighted as warm-colored areas. Our attack goal is to let the interpreters highlight the mid-left area of the image. In practice, such an attack goal can potentially mislead the end-users to believe that the mid-left area partially contributes to the prediction results.

From these visualization results, we can observe that the interpretations are shifted towards the desired interpretation. The highlight region is expanded, which means that the interpreter treats larger areas as important ones in terms of generating the classification results. Moreover, we also include one of the poisoning samples that we injected into the training set to manipulate the interpretation results. The poisoning samples look natural and similar to normal samples. Thus, these samples can be well camouflaged as benign samples and cannot be easily noticed at a glimpse.

5.3.3 Convergence Study. As presented in Section 4.2, the proposed learning algorithm (i.e., Algorithm 1) is an iterative procedure. Here we experimentally study the convergence property of the proposed algorithm. We report the changes of both the upper-level and the lower-level learning objectives on the Stanford Dogs dataset, i.e., $\mathcal{L}_{train}$ and $\mathcal{L}_{adv}$, and plot it with respect to the number of epoches in Figure 5. We can observe that the value of both objective functions becomes stable within dozens of epoches (i.e., about 15 epoches for upper-level and 20 epoches for lower-level). This phenomenon shows that the proposed learning algorithm converges nicely in practice.



(a) Upper Level Loss w.r.t Crafting Epoches



(b) Lower Level Loss w.r.t Crafting Epoches

Figure 5: Convergence Study of the Proposed Learning Algorithm

6 CONCLUSIONS

In this paper, we presented the first systematic study on the vulnerability of post-hoc outcome interpretations against data poisoning attacks. We proposed a bi-level optimization-based framework named IMF to generate adversarial poisoning samples for a variety of interpretation methods. The poisoning samples crafted by IMF encircle and shift the target sample in the feature space. As a result, the interpretation result of the target sample is manipulated as the attacker desires while keep the prediction performance. Nevertheless, solving the proposed bi-level optimization problem is challenging due to the computational bottleneck in the lower-level problem. To tackle the challenge, we proposed an efficient iterative learning algorithm and derived specific solutions against multiple representative interpretation methods. To evaluate the effectiveness

of the proposed framework, we conducted experiments on two real-world datasets against three representative interpretation models. Experimental results and case studies demonstrate the effectiveness of the proposed attack framework.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their helpful comments. This research was sponsored in part by the National Science Foundation IIS-1956017. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agency.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).
[2] Battista Biggio, Blaine Nelson, and Pavel Laskov. 2012. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389* (2012).
[3] Ginevra Carbone, Guido Sanguinetti, and Luca Bortolussi. 2021. Resilience of Bayesian Layer-Wise Explanations under Adversarial Attacks. *arXiv preprint arXiv:2102.11010* (2021).
[4] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. 2017. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526* (2017).
[5] Mark Craven and Jude W Shavlik. 1996. Extracting tree-structured representations of trained networks. In *Advances in neural information processing systems*. 24–30.
[6] Steven R Cummings, Dennis M Black, Michael C Nevitt, Warren S Browner, Jane A Cauley, Harry K Genant, Stephen R Mascioli, Jean C Scott, Dana G Seeley, Peter Steiger, et al. 1990. Appendicular bone density and age predict hip fracture in women. *Jama* 263, 5 (1990), 665–668.
[7] Piotr Dabkowski and Yarin Gal. 2017. Real time image saliency for black box classifiers. *arXiv preprint arXiv:1705.07857* (2017).
[8] Shihong Fang and Anna Choromanska. 2020. Backdoor Attacks on the DNN Interpretation System. *arXiv preprint arXiv:2011.10698* (2020).
[9] Ruth C Fong and Andrea Vedaldi. 2017. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*. 3429–3437.
[10] Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. 2017. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733* (2017).
[11] Satoshi Hara and Kohei Hayashi. 2016. Making tree ensembles interpretable. *arXiv preprint arXiv:1606.05390* (2016).
[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.
[13] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4700–4708.
[14] W Ronny Huang, Jonas Geiping, Liam Fowl, Gavin Taylor, and Tom Goldstein. 2020. Metapoison: Practical general-purpose clean-label data poisoning. *arXiv preprint arXiv:2004.00225* (2020).
[15] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. 2011. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*, Vol. 2. Citeseer.
[16] Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. 1885–1894.
[17] R Krishnan, G Sivakumar, and P Bhattacharya. 1999. Extracting decision trees from trained neural networks. *Pattern recognition* 32, 12 (1999).
[18] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*. 2980–2988.
[19] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. Ssd: Single shot multibox detector. In *European conference on computer vision*. Springer, 21–37.
[20] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*. 4765–4774.
[21] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).
[22] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017).
[23] Shike Mei and Xiaojin Zhu. 2015. Using machine teaching to identify optimal training-set attacks on machine learners. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.
[24] Luis Muñoz-González, Battista Biggio, Ambra Demontis, Andrea Paudice, Vasin Wongrassamee, Emil C Lupu, and Fabio Roli. 2017. Towards poisoning of deep learning algorithms with back-gradient optimization. In *Proc. of the 10th ACM Workshop on Artificial Intelligence and Security*. 27–38.
[25] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.
[26] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.
[27] Laura Rieger and Lars Kai Hansen. 2020. A simple defense against adversarial attacks on heatmap explanations. *arXiv preprint arXiv:2007.06381* (2020).
[28] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*. 618–626.
[29] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. 2018. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Advances in Neural Information Processing Systems*. 6103–6113.
[30] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).
[31] Mayank Singh, Nupur Kumari, Puneet Mangla, Abhishek Sinha, Vineeth N Balasubramanian, and Balaji Krishnamurthy. 2019. On the Benefits of Attributional Robustness. *arXiv. org* (2019).
[32] Ankur Sinha, Pekka Malo, and Kalyanmoy Deb. 2014. An improved bilevel evolutionary algorithm based on quadratic approximations. In *2014 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 1870–1877.
[33] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. 2017. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825* (2017).
[34] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. 2014. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806* (2014).
[35] Akshayvarun Subramanya, Vipin Pillai, and Hamed Pirsiavash. 2019. Fooling Network Interpretation in Image Classification. In *Proceedings of the IEEE International Conference on Computer Vision*. 2020–2029.
[36] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.
[37] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1–9.
[38] Heinrich Von Stackelberg and Stackelberg Heinrich Von. 1952. *The theory of the market economy*. Oxford University Press.
[39] Alexander Warnecke, Daniel Arp, Christian Wressnegger, and Konrad Rieck. 2020. Evaluating explanation methods for deep learning in security. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 158–174.
[40] Hengtong Zhang, Yaliang Li, Bolin Ding, and Jing Gao. 2020. Practical Data Poisoning Attack against Next-Item Recommendation. In *Proceedings of The Web Conference 2020*. 2458–2464.
[41] Hengtong Zhang, Tianhang Zheng, Jing Gao, Chenglin Miao, Lu Su, Yaliang Li, and Kui Ren. 2019. Data Poisoning Attack against Knowledge Graph Embedding. In *International Joint Conference on Artificial Intelligence*.
[42] Xinyang Zhang, Ningfei Wang, Hua Shen, Shouling Ji, Xiapu Luo, and Ting Wang. 2020. Interpretable deep learning under fire. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*.
[43] Tianhang Zheng and Baochun Li. 2021. First-Order Efficient General-Purpose Clean-Label Data Poisoning. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE.
[44] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. 2016. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2921–2929.