Neurocomputing 532 (2023) 56–66

Contents lists available at ScienceDirect

Neurocomputing

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

Depth-2 neural networks under a data-poisoning attack

Sayar Karmakar [a], Anirbit Mukherjee [b,1,*], Theodore Papamarkou [c,1]

[a] Department of Statistics, University of Florida, 230 Newell Drive, Gainesville 32611, FL, USA
[b] Department of Computer Science, The University of Manchester, Kilburn Building, Oxford Road, Manchester M139PL, UK
[c] Department of Mathematics, The University of Manchester, Alan Turing Building, Oxford Road, Manchester M13 9PL, UK

ARTICLE INFO

Article history:
Received 1 July 2022
Revised 21 December 2022
Accepted 16 February 2023
Available online 21 February 2023
Communicated by Zidong Wang

Keyword:
Convolutional neural networks
Stochastic algorithms
Data poisoning
Robust regression

ABSTRACT

In this work, we study the possibility of defending against data-poisoning attacks while training a shallow neural network in a regression setup. We focus on doing supervised learning with realizable labels for a class of depth-2 finite-width neural networks, which includes single-filter convolutional networks. In this class of networks, we attempt to learn the true network weights generating the labels in the presence of a malicious oracle doing stochastic, bounded and additive adversarial distortions on the true labels, during training. For the gradient-free stochastic algorithm that we construct, we prove worst-case near-optimal trade-offs among the magnitude of the adversarial attack, the weight approximation accuracy, and the confidence achieved by the proposed algorithm. As our algorithm uses mini-batching, we analyze how the mini-batch size affects convergence. We also show how to utilize the scaling of the outer layer weights to counter data-poisoning attacks on true labels depending on the probability of attack. Lastly, we give experimental evidence demonstrating how our algorithm outperforms stochastic gradient descent under different input data distributions, including instances of heavy-tailed distributions.

1. Introduction

The seminal paper by Szegedy et al. [1] was among the first to highlight a key vulnerability in state-of-the-art neural network architectures such as GoogLeNet, that adding small imperceptible adversarial noise to test data can dramatically impact the performance of the network. In these cases, despite the vulnerability of the predictive models to the distorted input, human observers are still able to correctly classify adversarially corrupted data.

In the last few years, experiments with adversarially attacked test data have been replicated on several state-of-the-art neural network implementations [2–5]. This phenomenon has also resulted in new adversarial defenses being proposed to counter the attacks. Such empirical observations have been systematically reviewed in Akhtar and Mian [6], Qiu et al. [7].

An optimization formulation of adversarial robustness, in terms of adversarial risk minimization on the test data, has been extensively explored in recent years; multiple attack strategies have been systematically catalogued in Dou et al. [8], Lin et al. [9], Song et al. [10], computational hardness of finding an adversarial risk minimizing hypothesis has been analyzed in Bubeck et al. [11], Degwekar et al. [12], Schmidt et al. [13], Montasser et al. [14], the issue of certifying adversarial robustness of a given predictor has been analyzed in Raghunathan et al. [15,16], and bounds on the Rademacher complexity of adversarial risk have been explored in Yin et al. [17], Khim and Loh [18].

On the other hand, the case of data-poisoning or adversarially attacked training data [19–22] has received much less attention from theoreticians - and in this work, we take some steps towards bridging that gap. Also, while these previous works on test data attacks have often been tuned to classification tasks, we consider the less explored case of adversarial attacks to neural networks used for regression tasks. In particular, we address a non-trivial challenge in this domain, namely we construct a training algorithm with poly-time convergence guarantee, which discovers a good approximation to the global minima of the unperturbed neural regression problem even when an adversary conducts online malicious distortions to the true labels used in training. We note that it is possible to construct training data attacks such that it would be information theoretically impossible to recover arbitrarily accurate approximations of the original global minima. Thus the challenge is to exploit as much structure as possible about the nature of the attack so that the recovered weights are nearly optimal for the worst attack allowed. This is exactly what we achieve in the

* Corresponding author.
[1] Equal Contribution.

https://doi.org/10.1016/j.neucom.2023.02.034
0925-2312/

setup of neural net class, data and adversarial attack that we consider.

Towards this end, firstly we note that the algebraic structure of depth-2 single filter convolution neural networks allow for a worst-case near-optimal defense against label poisoning during regression. Further, we realize that there is an appropriate generalization of these neural networks such that the 'filters' do not have to be 0/1 entry matrices (as are standard for convolutions), but can be arbitrary matrices which need only satisfy certain non-degeneracy conditions. In Definition 1 we formally specify the class of neural networks under consideration, and in Theorem 1 we specify the non-degeneracy condition.

For the optimization algorithm, we draw inspiration from the different versions of iterative stochastic gradient-free algorithms analyzed in the past [23–29]. A key insight that arises from these studies is that for certain neural losses, the usual gradient-based direction of update used in the training of the loss function of a neural network can be simplified without hurting performance, while allowing for a rigorous mathematical formalism. More specifically, the key simplification is to replace the 'derivative of activation' terms in the gradient by a linear function of the inputs. We allow for more flexible linearizations of this term (up to certain non-degeneracy conditions) than previous studies. Thus, we generalize this class of algorithms in the form of Algorithm 1. Despite the above simplification, the direction of update is not linear and is still a non-trivial function of the input data and the weights.

By allowing for arbitrary weights in the outer layer, we further expand the class of neural networks beyond the ambit of existing results. Subsequently, we run Algorithm 1 to train a neural network in the presence of an adversarial oracle that makes additive bounded perturbations to the true output of the network. Our theory establishes that there is a worst case near-optimal guarantee of recovery of true weights that our algorithm can achieve.

1.1. Related work

The existing studies of data-poisoning attacks on neural training have mostly focused on the classification setting. The major kinds of data-poisoning attacks on classifiers and attempts at defending against them can be grouped into three categories. Firstly, in backdoor attacks, the adversary injects strategically manipulated data [30]. Defence mechanisms against backdoor attacks have been proposed in Liu et al. [31], Tran et al. [32]. Secondly, in clean label attacks, the adversary does not modify the labels of the corrupted data [33,34]. Thirdly, in label flip attacks, the adversary changes the labels of a constant fraction of the data [35–37]. In the case of Massart noise, a robust half-space learning algorithm was analyzed in Diakonikolas et al. [38]. For more general predictors, the idea of randomized smoothing [39] has recently been extended and empirically shown to be capable of getting classifiers which are pointwise certifiably robust to label flip attacks [40].

Specifically for the setup of regression, to the best of our knowledge previous guarantees on achieving robustness against data-poisoning attacks have been limited to linear functions. Further, they have either considered corruptions that are limited to a small subset [41,36] of the input space or have made structural assumptions on the corruption. Despite the substantial progress with understanding robust linear regression [42–49], the corresponding questions have remained open even for simple neural networks.

Theoretical progress in understanding the limitations of training of a neural network under adversarial attacks on test data or training data has been restricted to deep kernel learning, which is associated with asymptotically large networks [50,51]. Developments along these lines have been made by Wang et al. [52], where the performance of stochastic gradient descent (SGD) is theoretically established when using it to train asymptotically large neural networks to perform classification in the presence of data-poisoning attacks.

Thus, it has been an open challenge to demonstrate an example of provably robust training when (a) the trained neural network is finite, (b) the training algorithm has the common structure of being iterative and stochastic, and (c) the training data is being adversarially attacked.

In this work, we take a few steps towards this goal by developing a framework inspired by the causative attack model [53,54].[2]

1.2. Summary of results and outline

The model of data-poisoning that we consider includes an additive distortion of the true output. For every data point, the additive distortion is a sample from a possibly different distribution. On the other hand, a typical model with additive noise relies on the assumption that all additive distortions are sampled from a single distribution [29]. Thus, the assumption of varying distribution of distortions is what makes a data-poisoning model distinct from a typical model of noisy output based on additive noise.

Our adversarial neural network herein is a multi-gate generalization of the single-gate model in [55]. Furthermore, we allow adversarial attacks to the multi-gate model that are capable of preventing optimal learning, as evidenced by our lower bound on the achievable accuracy of recovering the model parameters from the intact data (Section 3.5).

The main result (Theorem 1) shows that our proposed Algorithm 1 achieves a trade-off between accuracy, confidence and maximum allowed adversarial perturbation while learning the neural network parameters. This trade-off provides a performance guarantee that holds (i) in the presence of finitely many gates, (ii) under an online adversarial attack that does not assume access to the whole training data at the start of training, and (iii) for any probability of attack. To the best of our knowledge, there does not exist in the literature of neural network training any other such performance guarantee with all these three conditions being simultaneously true.

The parameter recovery accuracy of Algorithm 1, as guaranteed in Theorem 1, has two salient features. Firstly, our defense can defeat an adversarial attack in some scenarios, in the sense that the risk of the learnt predictor can be lower than the maximum adversarial distortion (Section 3.4). Secondly, the accuracy of parameter recovery improves by upscaling the weights of the second layer (Section 3.6).

In Section 4, we provide empirical evidence that our Algorithm 1 attains higher parameter recovery accuracy and faster rate of convergence than SGD. While a proof of this claim remains an open research question, our simulation-based observations are consistent across different input data distributions, probabilities of adversarial attack and magnitudes of adversarial attack. We also draw the attention of the reader to the experiment in Fig. 3 showing that Neuro-Tron outperforms SGD even when the data is sampled from the Student's t distribution with $\nu = 4$ degrees of freedom, which does not have enough number of finite moments to be covered by the assumptions of the main Theorem 1. This experiment in particular strongly motivates exciting directions of future research.

[2] The learning task in the causative attack model is framed as a game between a defender who seeks to learn and an attacker who aims to prevent this. In a typical scenario, the defender draws a finite number of samples from the true distribution (say S_c) and for some $\epsilon \in (0,1)$ the attacker mixes into the training data a set S_p of (maybe adaptively) corrupted training samples such that $|S_p| = \epsilon|S_c|$. Now the defender has to train on the set $S_p \cup S_c$. We note that our model is not the same as the causative attack model because we allow for an arbitrary fraction (including all) of the training data to be corrupted in an online fashion by the bounded additive adversarial attack on the true output.

Outline of the paper In Section 2, we give the mathematical setup of the neural networks, distributions and data-poisoning adversary that we use, and define our learning algorithm (Algorithm 1). In Section 3, we state our main result, which pertains to the parameter recovery accuracy of Algorithm 1, as Theorem 1. The proof of Theorem 1 is given in Appendix A, while appendices B–E contain several lemmas that are needed in the proof. In Section 3.5, we explain that the accuracy-confidence-attack trade-off we obtain is nearly optimal in the worst-case. In Section 3.6, we propose a way of improving the accuracy of parameter recovery for Algorithm 1 by upscaling the outer layer weights. In Section 4, we perform a simulation study of Algorithm 1, among else comparing it with SGD. We conclude in Section 5 by motivating relevant directions of future research.

2. Mathematical setup

In a supervised learning setting, consider observed data pairs $\mathbf{z} = (\mathbf{x}, y) \in \mathscr{Z} := \mathscr{X} \times \mathscr{Y}$, where $\mathbf{x}$ is the input in a measure space $\mathscr{X}$ and y is the output in a measure space $\mathscr{Y}$. Let $\mathscr{D}$ be the distribution of $\mathbf{z}$ over $\mathscr{Z}$ and let $\mathscr{D}_{\mathbf{x}}$ be the marginal distribution of the input x over $\mathscr{X}$. Consider a loss function $\ell : \mathscr{H} \times \mathscr{Z} \to \mathbb{R}^+$, where $\mathscr{H}$ is the hypothesis space for a learning task performed by a neural network. We model an adversarial oracle as a map $\boldsymbol{O_A} : \mathscr{Z} \mapsto \mathscr{Z}$ that corrupts the data $\mathbf{z} \in \mathscr{Z}$ with the intention of impeding the learning task. The uncorrupted data $\mathbf{z} \sim \mathscr{D}$ are not observed, so they are treated as a latent variable. Only the corrupted data $\boldsymbol{O_A}(\mathbf{z})$ are observed. The aim is to find a hypothesis $h \in \mathscr{H}$ that minimizes the true risk $\mathscr{R}(h; \mathscr{D}) := \mathbb{E}_{\mathbf{z}\sim\mathscr{D}}[\ell(h, z)]$.

In this work, we consider the case of input space $\mathscr{X} = \mathbb{R}^n$, output space $\mathscr{Y} = \mathbb{R}$, square loss function l and hypothesis space $\mathscr{H}$ of the class $\mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}}$ of depth-2 width-k neural networks. The class $\mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}}$ is specified in Definition 1.

Definition 1 (*Single-filter neural networks of depth 2 and width k*). Given a set of k sensing matrices $\mathscr{A} = \{\mathrm{A}_i \in \mathbb{R}^{r\times n} | i = 1, \ldots, k\}$, an α-leaky *ReLU* activation mapping $\sigma(y) = y\mathbf{1}_{y\geqslant 0} + \alpha y\mathbf{1}_{y<0}$ and a filter space $\mathscr{W} \subseteq \mathbb{R}^r$, we define the function class $\mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}}$ as

$$\mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}} := \left\{ f_{\mathbf{w}} : \mathbb{R}^n \to \mathbb{R} \text{with} f_{\mathbf{w}}(\mathbf{x}) = \frac{1}{k}\sum_{i=1}^{k} \sigma\left(\mathbf{w}^\top \mathrm{A}_i \mathbf{x}\right) | \mathbf{w} \in \mathscr{W} \right\}.$$

Note that the above class of neural networks encompasses the following common instances; (a) single *ReLU* gates as $\mathscr{F}_{1,0,\{\mathrm{I}_{n\times n}\},\mathbb{R}^n}$; (b) depth-2 width-$k$ convolutional neural networks, when each sensing matrix A_i has exactly one 1 in each row, at most one 1 in each column, and 0 in all other entries.

Training labels and the stochastic oracle that corrupts them We assume that $\exists \mathbf{w}^* \in \mathbb{R}^r$ such that each "true" data point $(\mathbf{x}, y)$ is s.t $y = f_{\mathbf{w}^*}(\mathbf{x})$. The oracle decides whether to attack a given data point $(\mathbf{x}, y)$ by performing a Bernoulli trial $\alpha_{\mathbf{x}} \sim \text{Bernoulli}(\beta(\mathbf{x}))$ with probability of attack $\beta(\mathbf{x}) := \Pr(\alpha_{\mathbf{x}} = 1)$. If $\alpha_{\mathbf{x}} = 1$, then the oracle replaces the original output $y = f_{\mathbf{w}^*}(\mathbf{x})$ with $f_{\mathbf{w}^*}(\mathbf{x}) + \xi_{\mathbf{x}}$, where $|\xi_{\mathbf{x}}| \leqslant \theta$ for some fixed θ. If $\alpha_{\mathbf{x}} = 0$, then the oracle returns the original output y without perturbing it. To ease notation, the adversarial output is denoted by $v := f_{\mathbf{w}^*}(\mathbf{x}) + \alpha_{\mathbf{x}}\xi_{\mathbf{x}}$. In summary, the adversarial action can be written as the map,

$$\boldsymbol{O_A}(\mathbf{x}, y) = \boldsymbol{O_A}(\mathbf{x}, f_{\mathbf{w}^*}(\mathbf{x})) = (\mathbf{x}, f_{\mathbf{w}^*}(\mathbf{x}) + \alpha_{\mathbf{x}}\xi_{\mathbf{x}}) = (\mathbf{x}, v).$$

While the above attack is happening on the training labels, the risk minimization problem is to find $\text{argmin}_{\mathbf{w}\in\mathscr{W}} \mathbb{E}_{\mathbf{x}\sim\mathscr{D}_{\mathbf{x}}}\left[\left(f_{\mathbf{w}^*}(\mathbf{x}) - f_{\mathbf{w}}(\mathbf{x})\right)^2\right]$. Lastly, we make the following Assumptions 1 and 2 for the distribution $\mathscr{D}_{\mathrm{x}}$ of input $\mathbf{x}$.

Assumption 1 (*Parity symmetry*). We assume that the distribution $\mathscr{D}_{\mathbf{x}}$ of the input $\mathbf{x}$ is symmetric under the parity transformation, i.e if $\mathbf{x}$ is a random variable such that $\mathbf{x} \sim \mathscr{D}_{\mathbf{x}}$ and $-\mathbf{x} \sim \mathscr{D}_{\mathbf{x}}$.

Assumption 2 (*Finiteness of first four moments of input norm*). We assume that the following expectations are finite,

$$\mathrm{m}_i := \mathbb{E}_{\mathbf{x}}\left[\|\mathbf{x}\|^i\right], i = 1, 2, 3, 4,$$

where $\|\cdot\|$ denotes the Euclidean norm thereafter. We note that for a measurable function $\beta : \mathbb{R}^n \to [0, 1]$, Assumption 2 implies finiteness of the expectations

$$\beta_i := \mathbb{E}_{\mathbf{x}}\left[\beta(\mathbf{x})\|\mathbf{x}\|^i\right], i = 1, 2, 3, 4.$$

$\beta(\mathbf{x})$ induces bias in the coin that the adversarial oracle tosses to decide whether or not to attack the true output y associated with input $\mathbf{x}$. To ease exposition, we introduce the notation

$$\bar{\mathrm{A}} := \frac{1}{k}\sum_{i=1}^{k} \mathrm{A}_i,\ \Sigma := \mathrm{E}[\mathbf{x}\mathbf{x}^\top],$$

$$\lambda_1 := \lambda_{\min}\left(\bar{\mathrm{A}}\Sigma\mathbf{M}^T\right),\ \lambda_2 := \sqrt{\lambda_{\max}\left(\mathbf{M}^T\mathbf{M}\right)},\ \lambda_3 := \frac{1}{k}\sum_{i=1}^{k}\lambda_{\max}\left(\mathrm{A}_i\mathrm{A}_i^\top\right),$$

where $\lambda_{\min}$ and $\lambda_{\max}$ denote the minimum and maximum eigenvalue, respectively, and $\mathbf{M} \in \mathbb{R}^{r\times n}$ is a "sensing matrix".

Algorithm 1 summarizes the proposed neural network training procedure in the presence of the assumed adversarial oracle. We refer to Algorithm 1 as the Neuro-Tron. Neuro-Tron is a stochastic optimization algorithm that does not use gradients, and it is inspired by Kakade et al. [26], Klivans and Meka [27], Goel et al. [29].

Algorithm 1: Neuro-Tron (mini-batched, multi-gate, single-filter, stochastic algorithm)

Input: Sampling access to the marginal input distribution $\mathscr{D}_{\mathbf{x}}$
Input: Access to adversarial output $v \in \mathbb{R}$ for any input $\mathbf{x} \in \mathbb{R}^n$
Input: Access to output $f_{\mathbf{w}}(\mathbf{x})$ of any $f_{\mathbf{w}} \in \mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}}$ for any $\mathbf{w} \in \mathbb{R}^r$ and any input $\mathbf{x}$
Input: A sensing matrix $\mathbf{M} \in \mathbb{R}^{r\times n}$
Input: A starting point $\mathbf{w}_1$
Input: Number T of batches
Input: Batch size b
Input: Learning rate η
for $t = 1, \ldots, \mathrm{T}$ **do**
  Sample batch $s_t := (\mathbf{x}_{t_1}, \ldots, \mathbf{x}_{t_b})$, where $\mathbf{x}_{t_i} \sim \mathscr{D}_{\mathbf{x}}, i = 1, \ldots, b$
  for $i = 1, \ldots, b$ **do**
    The oracle samples $\alpha_{t_i} \in \{0, 1\}$ with probability $\{1 - \beta(x_{t_i}), \beta(x_{t_i})\}$
    The oracle replies with $v_{t_i} := f_{\mathbf{w}^*}(\mathbf{x}_{t_i}) + \alpha_{t_i}\xi_{t_i}$
  end for
  Form the so-called Tron-gradient,

$$\mathbf{g}^{(t)} := \mathbf{M}\left(\frac{1}{b}\sum_{i=1}^{b}((v_{t_i} - f_{\mathbf{w}^{(t)}}(\mathbf{x}_{t_i}))\mathbf{x}_{t_i})\right)$$

  $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \eta\mathbf{g}^{(t)}$
end for

3. Probabilistic performance bounds

This section states the main theorem (Theorem 1), which provides the optimal learning rate for Algorithm 1 to approximate the neural network weights at a given accuracy level ϵ with a given failure probability δ. To illustrate the consequences of Theorem 1, the special case of normally distributed input data is discussed.

3.1. The main theorem

Theorem 1 (*Trade-off between accuracy and failure*). Let $f_{\mathbf{w}}$ be a neural network belonging to the function class $\mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}}$ of Definition 1. We assume that there exists $\mathbf{w}^* \in \mathbb{R}^r$ such that for each input $\mathbf{x}$ sampled from $\mathscr{D}_{\mathbf{x}}$ there is a corresponding true label $y = f_{\mathbf{w}^*}(\mathbf{x})$. An adversarial oracle corrupts the true label y by generating the label $f_{\mathbf{w}^*}(\mathbf{x}) + \alpha_{\mathbf{x}}\xi_{\mathbf{x}}$, where $\xi_{\mathbf{x}}$ is chosen to satisfy $|\xi_{\mathbf{x}}| \leqslant \theta$ for some fixed θ, and $\alpha_{\mathbf{x}} \sim \text{Bernoulli}(\beta(\mathbf{x}))$ for some measurable function $\beta : \mathbb{R}^n \to [0,1]$. Thus, the oracle sends the tuple $\boldsymbol{O}_{\boldsymbol{A}}(\mathbf{x}, y) = (\mathbf{x}, f_{\mathbf{w}^*}(\mathbf{x}) + \alpha_{\mathbf{x}}\xi_{\mathbf{x}})$. Additionally, we assume that Assumptions 1 and 2 are satisfied, and that $\lambda_i > 0, i = 1,2,3$.

1. Assume that $\theta = 0$, which corresponds to the case of uncorrupted adversarial action $\boldsymbol{O}_{\boldsymbol{A}}(\mathbf{x}, y) = (\mathbf{x}, f_{\mathbf{w}^*}(\mathbf{x}))$. If the learning rate η in Algorithm 1 is equal to
$$\eta = \frac{\lambda_1}{\gamma(1+\alpha)\lambda_2^2\lambda_3\left(\mathrm{m}_4/b + \mathrm{m}_2^2(1-1/b)\right)},$$
where $\gamma > \max\{C, 1\}$ with $C := \lambda_1^2\left(\lambda_2^2\lambda_3\left(\mathrm{m}_4/b + \mathrm{m}_2^2(1-1/b)\right)\right)^{-1}$, then for accuracy $\epsilon \geqslant 0$, for failure probability $\delta \geqslant 0$, and for $\mathrm{T} = \mathcal{O}\left(\log\left(\frac{\|\mathbf{w}^{(1)} - \mathbf{w}^*\|^2}{\epsilon^2\delta}\right)\right)$ it holds that $\|\mathbf{w}^{(\mathrm{T})} - \mathbf{w}^*\| \leqslant \epsilon$ with probability at least $1 - \delta$.
2. Assume that $\theta \in (0, \theta_*)$ for some $\theta_* > 0$, which corresponds to the case of adversarial perturbation via additive noise. We assume that β_1 is such that the constant $\mathrm{c}_{\text{trade-off}} := \frac{(1+\alpha)\lambda_1}{\beta_1\lambda_2} - 1 > 0$. Moreover, assume that the distribution $\mathscr{D}_{\mathbf{x}}$, matrix $\mathbf{M}$ in Algorithm 1, noise bound θ_*, target accuracy ϵ and target confidence $\delta > 0$ are such that
$$\theta_*^2 = \epsilon^2\delta\mathrm{c}_{\text{trade-off}}, \epsilon^2\delta < \|\mathbf{w}^{(1)} - \mathbf{w}^*\|^2. \tag{1}$$
If the learning rate η in Algorithm 1 is equal to
$$\eta = \frac{\beta_1\mathrm{c}_{\text{trade-off}}}{\gamma(1+\alpha)^2\lambda_2\lambda_3\left(\left(\beta_1\mathrm{m}_2 + \mathrm{m}_2^2\right)\left(1-\frac{1}{b}\right) + \frac{\beta_3+\mathrm{m}_4}{b}\right)},$$
where
$$\gamma > \max\left\{\frac{\left(\beta_1\mathrm{c}_{\text{trade-off}}\right)^2}{(1+\alpha)^2\lambda_3\left(\left(\beta_1\mathrm{m}_2 + \mathrm{m}_2^2\right)\left(1-\frac{1}{b}\right) + \frac{\beta_3+\mathrm{m}_4}{b}\right)}, C_2\right\} > 1$$
with
$$C_2 := \frac{\epsilon^2\delta + \frac{\theta^2\left(\left(\beta_1^2+\beta_1\mathrm{m}_2\right)\left(1-\frac{1}{b}\right) + \frac{\beta_2+\beta_3}{b}\right)}{(1+\alpha)^2\lambda_3\left(\beta_1\mathrm{m}_2+\mathrm{m}_2^2\right)\left(1-\frac{1}{b}\right)+\frac{\beta_3+\mathrm{m}_4}{b}}}{\epsilon^2\delta - \frac{\theta^2}{\mathrm{c}_{\text{trade-off}}}},$$
then for
$$\mathrm{T} = \mathcal{O}\left(\log\left[\frac{\|\mathbf{w}^{(1)} - \mathbf{w}^*\|^2}{\epsilon^2\delta - \frac{\theta^2}{\mathrm{c}_{\text{rate}}}}\right]\right)$$
with
$$\mathrm{c}_{\text{rate}} := \frac{\gamma - 1}{\frac{\mathrm{m}_2+\mathrm{m}_3}{(1+\alpha)^2\lambda_3(\mathrm{m}_3+\mathrm{m}_4)} + \frac{\gamma}{\mathrm{c}_{\text{trade-off}}}}$$
it holds that $\|\mathbf{w}^{(\mathrm{T})} - \mathbf{w}^*\| \leqslant \epsilon$ with probability at least $1 - \delta$.

Remark 1. Some remarks on Theorem 1 follow.

1. Theorem 1 places weak conditions, which can be easily met in practice. Firstly, it is easy to find a distribution $\mathscr{D}_{\mathbf{x}}$ that satisfies Assumptions 1 and 2 and that has a positive definite covariance matrix Σ. Secondly, for any full rank $\mathbf{M} \in \mathbb{R}^{r\times n}$, any matrix $\mathrm{C} \in \mathbb{R}^{r\times n}$ and any even width w (say $w = 2k$), the sensing matrices in Definition 1 can be set to $\mathscr{A} = \{(\mathbf{M} - k\mathrm{C}, \mathbf{M} - (k-1)\mathrm{C}, \ldots, \mathbf{M} - \mathrm{C}, \mathbf{M} + \mathrm{C}, \ldots, \mathbf{M} + k\mathrm{C}\}$. Then $\overline{A} = \mathbf{M}$ has full rank, so $\lambda_1 = \lambda_{\min}(\mathbf{M}\Sigma\mathbf{M}^\top) > 0$ as required. Thirdly, it is easy to construct a sampling scheme that generates a matrix $\mathbf{M} \in \mathbb{R}^{r\times n}$ with $1 \leqslant r \leqslant n$ which is full rank with high probability. To this end, generate independent $\mathbf{g}_i = \left(g_i^1, \ldots, g_i^r\right)^\top \sim N(0, \mathrm{I}_{r\times r})$ and construct $G = \sum_{i=1}^k \mathbf{g}_i\mathbf{g}_i^\top$. Then G follows a Wishart distribution $\mathbb{W}(\mathrm{I}, k)$ with k degrees of freedom. Since I is invertible, G has full rank with probability 1 as long as $k \geqslant r$. G can be used as a sub-matrix to complete it as a matrix $\mathbf{M} \in \mathbb{R}^{r\times n}$ which also has full rank with probability 1. Lastly, consider the case when $\beta(\mathbf{x})$ is a constant β. Then we note that the condition of $c_{\text{trade-off}}$ being positive is equivalent to $\beta < \frac{(1+\alpha)\lambda_{\min}\left(\bar{\mathrm{A}}\Sigma\mathbf{M}^\top\right)}{\mathbb{E}[\|\mathbf{x}\|]\cdot\|\mathbf{M}\|_2}$. From here we can see that if we anticipate a large probability β of attack, we can scale the vectors in the support of distribution $\mathscr{D}_{\mathbf{x}}$ by an appropriate positive factor and enlarge the upper bound for β by the same factor.
2. The uniqueness of the global minimum for $\theta = 0$ can be proven by contradiction. Assume that there are two distinct minima $\text{argmin}_{\mathbf{w}\in\mathscr{W}}\mathbb{E}_{\mathbf{x}\sim\mathscr{D}_{\mathbf{x}}}\left[\left(f_{\mathbf{w}^*}(\mathbf{x}) - f_{\mathbf{w}}(\mathbf{x})\right)^2\right]$. The application of Theorem 1 to each minimum separately implies that Algorithm 1 gets arbitrarily close to each minimum, which is a contradiction.
3. In both cases $\theta = 0$ and $\theta \in (0, \theta_*)$, the learning rate η is an increasing function of the mini-batch size b. So increasing b increases the rate of convergence.
4. In the case of $\theta \in (0, \theta_*)$, the term $\epsilon^2\delta - \frac{\theta^2}{c_{\text{rate}}}$ in the expression of T is positive because of the lower bound imposed on the parameter γ.
5. In SubSection 3.5, we show that the trade-off between optimization accuracy and failure probability is near-optimal in the worst-case scenario, that is when the adversary attacks every data point.

3.2. Sketch of the proof of the main theorem

The proof of Theorem 1 is given in Appendices A, B, C and E. An outline of the proof follows. Initially, the proof disentangles the dependencies between random variable $\mathbf{g}^{(t)}$, sampled data s_t and coin flips $a_t := (\alpha_{t_1}, \ldots, \alpha_{t_b})$ that determine whether or not to attack the corresponding output data. Let $s_{1:t} := (s_1, \ldots, s_t)$ be the training data sampled by Algorithm 1 till the t-th iteration. The neural network weights $\mathbf{w}_t$ at time t are dependent conditional on $s_{1:(t-1)}$. The random variable g_t is dependent on s_t, on α_t, and on $(\xi_{t_1}, \ldots, \xi_{t_b})$. The key idea in the proof is to find a tight upper bound on the random variables
$$\mathbb{E}_{s_t,\alpha_t}\left[\|\mathbf{w}^{(t+1)} - \mathbf{w}^*\|^2 - \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 \,|\, s_{1:(t-1)}\right].$$
To acquire such an upper bound, we invoke Assumption 1 and we track the combinatorial effect of mini-batching. Finally, we take total expectations over the above upper bound and reduce the prob-

lem of finding convergence times to a problem of analyzing certain algebraic recursions. These recursions are established in the lemmas of Appendix E.

3.3. Performance bounds for normally distributed input data

To understand the constraints imposed by Eq. (1) of Theorem 1, we assume normally distributed input data and consider a single *ReLU* gate neural network $\mathscr{F}_{1,0,\{\mathbf{I}_{n\times n}\},\mathbb{R}^n}$. Lemma 1 provides the constant $c_{\text{trade-off}}$ under this setting.

Lemma 1. [Accuracy-failure trade-off for normally distributed input] Consider a single *ReLU* gate neural network $\mathscr{F}_{1,0,\{\mathbf{I}_{n\times n}\},\mathbb{R}^n}$. Assume that the input data $\mathbf{x}$ are normally distributed according to $\mathscr{D}_{\mathbf{x}} = \mathcal{N}(0,\sigma^2\mathbf{I}_{n\times n})$. If $\beta(\mathbf{x}) =: \beta \in (0,1)$ for all $\mathbf{x}$ and $\mathbf{M} = \mathbf{I}_{n\times n}$ in Algorithm 1, then the constant $c_{\text{trade-off}}$ in Theorem 1 is given by

$$c_{\text{trade-off}} = \frac{\theta_*^2}{\epsilon^2\delta} = \frac{\sigma}{\sqrt{2}\beta}\frac{\Gamma\left(\frac{n}{2}\right)}{\Gamma\left(\frac{n+1}{2}\right)} - 1. \tag{2}$$

The proof of Lemma 1 can be found in Appendix D. If the input data distribution is $\mathcal{N}(0,\sigma^2\mathbf{I})$, where σ^2 is an increasing function of the input data dimension n such that the right-hand side of Eq. (2) remaines fixed, then Eq. (2) provides a sufficient condition to defend against an adversary with a fixed corruption budget of θ_*, with a desired accuracy of ϵ and with failure probability of δ.

3.4. Understanding the prediction risk

The prediction risk of a neural network $\mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}}$ at time T is

$$\mathbb{E}_{\mathbf{x}\sim\mathscr{D}_{\mathbf{x}}}\left[\left(f_{\mathbf{w}^*}(\mathbf{x}) - f_{\mathbf{w}^{(\mathrm{T})}}(\mathbf{x})\right)^2\right] = \mathbb{E}_{\mathbf{x}\sim\mathscr{D}_{\mathbf{x}}}\left[\left(\frac{1}{k}\sum_{i=1}^{k}\left\{\sigma\left(\mathbf{w}^{*\top}\mathrm{A}_i\mathbf{x}\right) - \sigma\left(\mathbf{w}^{(\mathrm{T})\top}\mathrm{A}_i\mathbf{x}\right)\right\}\right)^2\right].$$

As shown in Lemma 3 (Appendix C), if the conditions of Eq. (1) of Theorem 1 are satisfied and if the upper bound

$$\left(\frac{(1+\alpha)^2}{k\delta c_{\text{trade-off}}}\sum_{i=1}^{k}\lambda_{\max}\left(\mathrm{A}_i\mathrm{A}_i^\top\right)\right)\mathbb{E}_{\mathbf{x}\sim\mathscr{D}_{\mathbf{x}}}\left[\|\mathbf{x}\|^2\right] < 1 \tag{3}$$

holds at iteration T, then the risk $\mathbb{E}_{\mathbf{x}\sim\mathscr{D}_{\mathbf{x}}}\left[\left(f_{\mathbf{w}^*}(\mathbf{x}) - f_{\mathbf{w}^{(\mathrm{T})}}(\mathbf{x})\right)^2\right]$ is bounded above by θ_*^2.

It is easy to demonstrate cases for which Inequality (3) holds. For example, the assumptions of Lemma 1 imply that Inequality (3) is equivalent to

$$n = \mathbb{E}_{\mathbf{x}\sim\mathscr{D}_{\mathbf{x}}}\left[\|x\|^2\right] \leqslant \delta\left(\frac{1}{\beta_1} - 1\right) \text{or} \beta_1 \leqslant \frac{1}{1+n/\delta} \tag{4}$$

in the case of a single *ReLU* gate neural network $\mathscr{F}_{1,0,\{\mathbf{I}_{n\times n}\},\mathbb{R}^n}$ trained on normally distributed input data. Eq. (D.1) and Inequality (4) yield the upper bound

$$\beta < \frac{1}{\sqrt{2}}\left[\frac{\Gamma\left(\frac{n}{2}\right)}{\Gamma\left(\frac{n+1}{2}\right)}\right]\frac{1}{1+n/\delta} \tag{5}$$

for the probability β of adversarial attack. Note that this bound on β depends on the input data dimension n. So, if the probability β of attack admits the upper bound of Inequality (5) while training a single *ReLU* gate neural network on normally distributed input data, then the learnt weights attain higher average prediction accuracy than the worst distortion the oracle could have made to any particular output data point.

3.5. Demonstrating near-optimality in the worst case

We recall that Case 1 of Theorem 1 ($\theta = 0$) shows that Algorithm 1 recovers the true filter $\mathbf{w}^* \in \mathbb{R}^r$ when it has access to the uncorrupted data. Consider a filter value $\mathbf{w}_{\mathbf{x}} \neq \mathbf{w}^*$ given the true filter $\mathbf{w}^*$, and suppose that $\theta_* = \zeta$ for some $\zeta \geqslant \sup_{\mathbf{x}\in\text{supp}(\mathscr{D}_{\mathbf{x}})}|f_{\mathbf{w}_{\text{adv}}}(x) - f_{\mathbf{w}^*}(x)|$, where $\text{supp}(\mathscr{D}_{\mathbf{x}})$ denotes the support of $\mathscr{D}_{\mathbf{x}}$. Assume that $\mathscr{D}_{\mathbf{x}}$ is compactly supported, so that the supremum exits. In this setting, Eq. (1) yields $\epsilon^2 \geqslant \zeta^2/c_{\text{trade-off}}$. Hence proving optimality of the guarantee is equivalent to showing the existence of an attack which satisfies the upper bound ζ of $\sup_{\mathbf{x}\in\text{supp}(\mathscr{D}_{\mathbf{x}})}|f_{\mathbf{w}_{\text{adv}}}(x) - f_{\mathbf{w}^*}(x)|$ and for which the best possible accuracy nearly saturates the lower bound $\zeta^2/c_{\text{trade-off}}$ of ϵ^2.

If the adversarial oracle $\mathbf{O}_{\text{A}}$ is queried at $\mathbf{x}$ under this choice of θ_*, then the oracle replies with $\xi_{\mathbf{x}} + f_{\mathbf{w}_*}(\mathbf{x})$, where $\xi_{\mathbf{x}} = f_{\mathbf{w}_{\text{adv}}}(\mathbf{x}) - f_{\mathbf{w}_*}(\mathbf{x})$. So, the data Algorithm 1 receives are exactly realized with filter $\mathbf{w}_{\text{adv}}$. Thus, Case 1 of Theorem 1 implies that Algorithm 1 converges to $\mathbf{w}_{\text{adv}}$ with high probability and with error $\|\mathbf{w}_{\text{adv}} - \mathbf{w}_*\| \leqslant \epsilon$.

We now consider the above attack happening to a single *ReLU* gate neural network $f_{\mathbf{w}_*}(\mathbf{x}) = ReLU(\mathbf{w}_*^\top\mathbf{x}), \mathbf{x} \in \mathbb{R}^n$, with $\zeta = r\|\mathbf{w}_{\text{adv}} - \mathbf{w}_*\|$, where $r := \sup_{\mathbf{x}\in\text{supp}(\mathscr{D}_{\mathbf{x}})}\|\mathbf{x}\|$. Assume that r is finite and that $\mathscr{D}_{\mathbf{x}}$ satisfies Assumptions 1 and 2. This choice of ζ is valid since the following holds,

$$\sup_{\mathbf{x}\in\text{supp}(\mathscr{D}_{\mathbf{x}})}|ReLU(\mathbf{w}_{\text{adv}}^\top\mathbf{x}) - ReLU(\mathbf{w}_*^\top\mathbf{x})| \leqslant r\|\mathbf{w}_{\text{adv}} - \mathbf{w}_*\| = \zeta.$$

Such a setup for training a single *ReLU* gate neural network on output data additively corrupted by at most $\zeta = r\|\mathbf{w}_{\text{adv}} - \mathbf{w}_*\|$ demonstrates a worst case scenario (i.e $\beta(\mathbf{x}) = 1$) in which the accuracy guarantee of $\epsilon^2 \geqslant \zeta^2/c_{\text{trade-off}}$ is optimal up to a constant $r^2/c_{\text{trade-off}}$. The near-optimality of Eq. (1) holds for any algorithm defending against this attack, if the algorithm has the property of recovering the parameters correctly when the output data are exactly realizable.

3.6. Defense against data-poisoning attacks via upscaled outer layer weights

Definition 2 generalizes the class of neural networks of Definition 1 by introducing a weighted sum of gates computed by a neural network in the class. The weights $q \in \mathscr{W}_2 \subseteq \mathbb{R}^k$ of the second layer play the role of weights in the sum of gates and augment the network parameter space $\mathscr{W}_1$ of Definition 1 by $\mathscr{W}_2$.

Definition 2 (*Weighted neural networks of depth 2 and width k*). Given k sensing matrices $\mathscr{A} = \{\mathrm{A}_i \in \mathbb{R}^{r\times n} | i = 1,\ldots,k\}$, an α-leaky *ReLU* activation mapping $\sigma(y) = y\mathbf{1}_{y\geqslant 0} + \alpha y\mathbf{1}_{y<0}$, a filter space $\mathscr{W}_1 \subseteq \mathbb{R}^r$ and a space of values for the second layer weights $\mathscr{W}_2 \subseteq \mathbb{R}^k$, we define the function class $\mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}_{1,2}}$ as

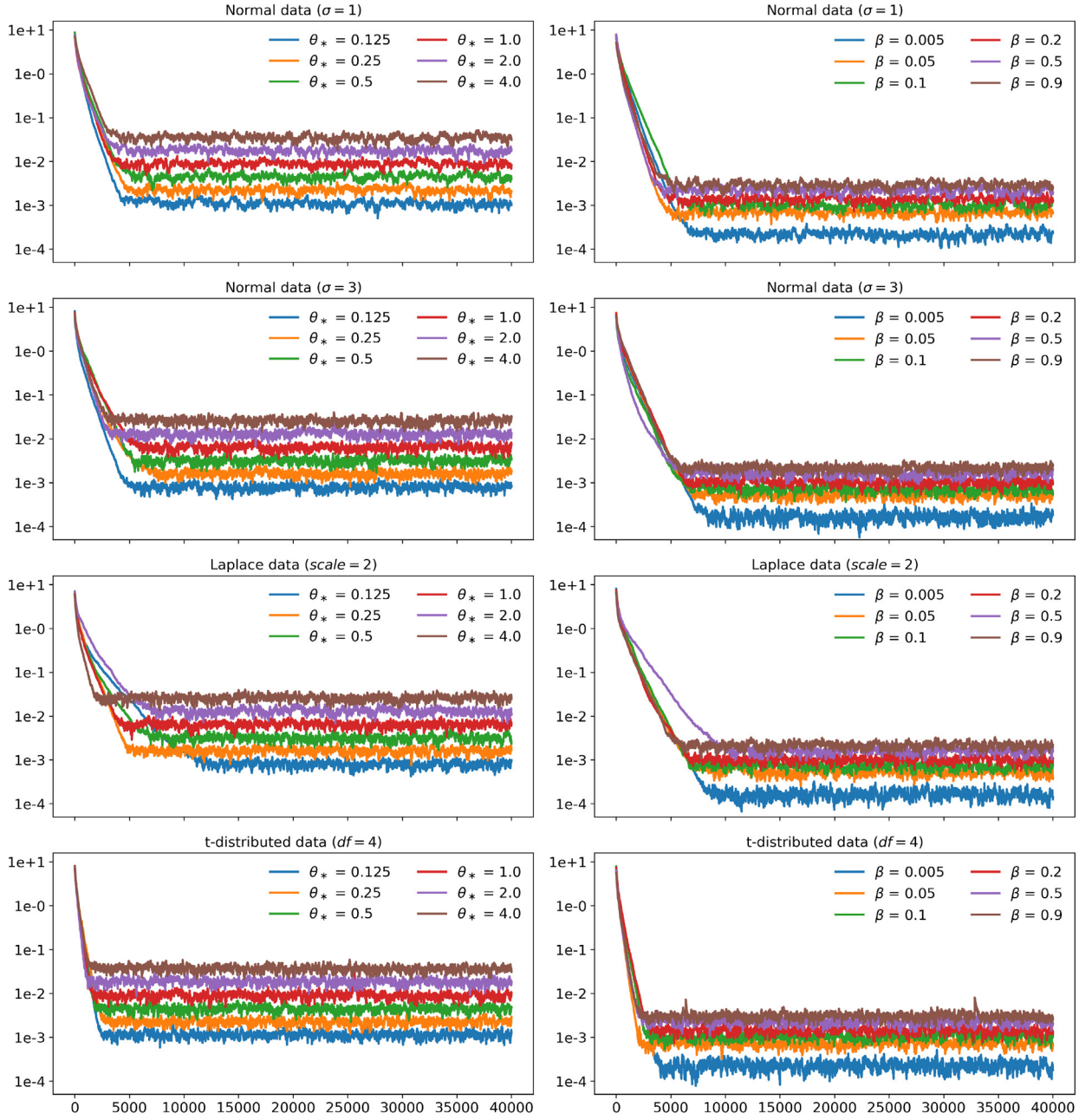
$$\mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}_{1,2}} := \left\{f_{\mathbf{q},\mathbf{w}} : \mathbb{R}^n \to \mathbb{R}, f_{\mathbf{q},\mathbf{w}}(\mathbf{x}) = \frac{1}{k}\sum_{i=1}^{k} q_i \cdot \sigma\left(\mathbf{w}^\top\mathrm{A}_i\mathbf{x}\right) \in \mathbb{R} | \mathbf{w} \in \mathscr{W}_1, \mathbf{q} \in \mathscr{W}_2\right\}.$$

The analysis in the proof of Theorem 1 is applicable when $f_{\mathbf{w}}$ is replaced by $f_{\mathbf{q},\mathbf{w}}$ for fixed $\mathbf{q}$. Consequently, Theorem 1 continues to hold for $\bar{\mathrm{A}}$ and λ_3 set to

$$\bar{\mathrm{A}} := \frac{1}{k}\sum_{i=1}^{k} q_i\mathrm{A}_i, \quad \lambda_3 := \frac{1}{k}\sum_{i=1}^{k} q_i^2\lambda_{\max}\left(\mathrm{A}_i\mathrm{A}_i^\top\right). \tag{6}$$

Table 1
Configuration of hyperparameters (rows) across eight experimental setups (columns). Each setup arises from a combination of an input data distribution and of a varying hyperparmater. The varying hyperparameter is either the noise bound $\theta_* \in \{0, 0.125, 0.25, 0.5, 1, 2, 4\}$ or the probability of adversarial attack $\beta \in \{0.005, 0.05, 0.1, 0.2, 0.5, 0.9\}$. The rest of hyperparameters are the following: learning rate η, input data dimension n, filter size r, batch size b, and network width k.

Data	$\mathcal{N}(0,1)$		t(4)		$\mathcal{N}(0,9)$		Laplace(0, 2)	
θ_*	Varying	0.25	Varying	0.25	Varying	0.25	Varying	0.25
β	0.5	Varying	0.5	Varying	0.5	Varying	0.5	Varying
η	0.0001	0.0001	0.0001	0.0001	0.00005	0.00005	0.00005	0.00005
n	100	100	100	100	50	50	50	50
r	25	25	25	25	25	25	25	25
b	16	16	16	16	16	16	16	16
k	10	10	10	10	10	10	10	10



(a) Different $\theta_\star$ values. (b) Different β values.

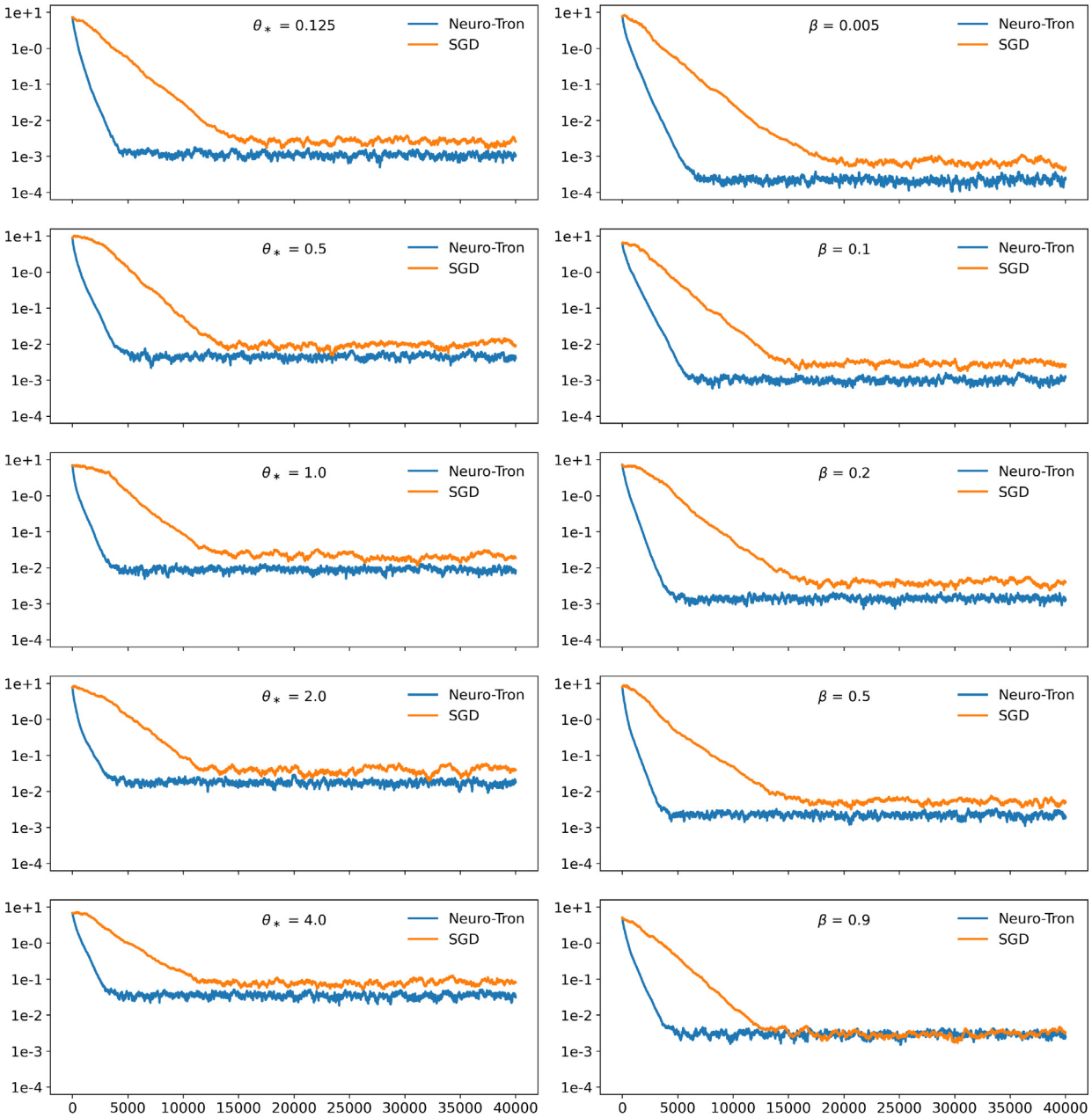
Fig. 1. Simulation-based validation of Theorem 1 regarding the performance of Algorithm 1 (Neuro-Tron). (a): Neuro-Tron parameter recovery errors per input data distribution for different adversarial noise bounds θ_*. (b): Neuro-Tron parameter recovery errors per input data distribution for different probabilities β of adversarial attack.

Eq. (6) sets the constraint of positive $\lambda_1 = \lambda_{\min}\left(\bar{\mathrm{A}}\Sigma\mathbf{M}^T\right)$ on $\mathbf{M}$.

Here we consider a special case of a neural network with a weighted sum of gates, satisfying Definition 2. By analyzing this special case we shall reveal an interesting insight about how weights in the outer layer of the network can help to defend against the attack being considered. Consider neural networks $f_{\mathbf{1},\mathbf{w}}$ in $\mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}_{1,2}}$ with sensing matrices $\mathrm{A}_i, i = 1,\ldots,k$, for the first layer of the network and note that $f_{\mathbf{1},\mathbf{w}} = f_{\mathbf{w}} \in \mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}}$. Set $\mathbf{M}$ such that $\lambda_1 = \lambda_{\min}\left(\bar{\mathrm{A}}\Sigma\mathbf{M}^T\right) > 0$, where $\bar{\mathrm{A}} = \frac{1}{k}\sum_{i=1}^{k}\mathrm{A}_i$. Further, given a real number $q \neq 0$, consider another class of neural networks $f_{\prime_{q^2\mathbf{1},\mathbf{w}}}$ in $\mathscr{F}_{k,\alpha,\mathscr{A},\mathscr{W}_{1,2}}$. Note that $\lambda_{1\prime} := \lambda_{\min}\left(\bar{\mathrm{A}\prime}\Sigma\mathbf{M}^T\right) = q^2\lambda_1 > 0$, where $\overline{A\prime} := \frac{q^2}{k}\sum_{i=1}^{k}\mathrm{A}_i$. Thus, $\mathbf{M}$ ensures convergence of Algorithm 1 while training over both network classes. Assume that the constants β_1 and θ_*, which characterize the adversarial attack, and the 'lack of confidence' δ are fixed. If ϵ and $\epsilon\prime$ are the guaranteed accuracies of recovering the true weights when $\mathbf{q} = \mathbf{1}$ and $\mathbf{q} = q^2\mathbf{1}$, respectively, the it follows from Eq. (1) that

$$\epsilon = \theta_*\sqrt{\frac{1}{\delta\left(\frac{(1+\alpha)\lambda_1}{\beta_1\lambda_2} - 1\right)}}, \epsilon\prime = \theta_*\sqrt{\frac{1}{\delta\left(q^2\cdot\frac{(1+\alpha)\lambda_1}{\beta_1\lambda_2} - 1\right)}}.$$

For this special case, we note that a multiplicative increase in β_1, by say $c_1 > 1$, can be compensated by letting the attack happen while training over neural networks $f_{c_1\mathbf{1},\mathbf{w}}$. Similarly, a multiplicative increase in θ_*, by say $c_2 > 1$, can be compensated by letting the attack happen while training over neural networks $f_{c_2\mathbf{1},\mathbf{w}}$, since



(a) Neuro-Tron versus SGD per $\theta_\star$ value. (b) Neuro-Tron versus SGD per β value.

Fig. 2. Simulation-based comparison between Algorithm 1 (Neuro-Tron) and SGD. Input data are sampled from $\mathscr{N}(\mu = 0, \sigma^2 = 1)$. (a): Parameter recovery errors for different adversarial noise bounds $\theta_\star$. (b): Parameter recovery errors for different probabilities β of adversarial attack.

$$c_2\theta_* \sqrt{\frac{1}{\delta\left(c_2^2 \cdot \frac{(1+\alpha)\lambda_1}{\beta_1\lambda_2} - 1\right)}} < \theta_* \sqrt{\frac{1}{\delta\left(\frac{(1+\alpha)\lambda_1}{\beta_1\lambda_2} - 1\right)}}.$$

If Algorithm 1 is used for training over neural networks $f_{\mathbf{1},\mathbf{w}}$ and $f_{\prime_{q^2}\mathbf{1},\mathbf{w}}, q > 1$, with the same $\{\mathrm{A}_i\}_{i=1}^k$ matrices, then one can choose a common $\mathbf{M}$ for both the instances such that while facing the same output-poisoning adversary, the accuracy of recovering the true weights in class $f_{\prime_{q^2}\mathbf{1},\mathbf{w}}$ is higher than the accuracy of recovering the true weights in class $f_{\mathbf{1},\mathbf{w}}$. In other words, increasing the outer layer weights via higher values of $q > 1$ improves the accuracy-related defence of Algorithm 1 against the same type of adversarial attack. To this end, we demonstrate via a simulation-based experiment the accuracy advantage gained by upscaling the outer layer weights (see Appendix H).

4. Simulation study

In this section, we conduct a simulation study by training *ReLU* neural networks on different input data distributions and for different hyperparameter settings of Algorithm 1. The purpose of the simulation study is twofold, to empirically validate the relative theoretical performance bounds of Algorithm 1 and to compare Algorithm 1 with SGD. The code for our simulations can be found at https://github.com/papamarkou/neurotron_experiments.

Eight setups are included in our simulation study. For each setup, independent and identically distributed input data samples $\mathbf{x}_{t_i}, i = 1, \ldots, b$, are drawn from one of the following four distributions: standard normal $\mathcal{N}(\mu = 0, \sigma^2 = 1)$, normal $\mathcal{N}(\mu = 0, \sigma^2 = 9)$ with mean $\mu = 0$ and variance $\sigma^2 = 9$,

(a) Neuro-Tron vs SGD per $\theta_\star$ value. (b) Neuro-Tron vs SGD per β value.

Fig. 3. Simulation-based comparison between Algorithm 1 (Neuro-Tron) and SGD. Input data are sampled from Student's t($\nu = 4$). (a): Parameter recovery errors for different adversarial noise bounds $\theta_\star$. (b): Parameter recovery errors for different probabilities β of adversarial attack.

Laplace($\mu = 0, b = 2$) with mean $\mu = 0$ and scale $b = 2$, or Student's t-distribution t($\nu = 4$) with $\nu = 4$ degrees of freedom. For each setup associated with an input data distribution, either the noise bound θ_* or the probability β of adversarial attack vary, while the remaining hyperparameters are fixed. To sum up, in each of the eights setups, one out of four possible distributions is selected to sample input data, and one of the two hyperparameters θ_* or β vary.

To run a simulation for a given setup, we initially sample a point $\mathbf{w}_* \sim \mathcal{N}(0, \mathrm{I}_r)$ and sample the sensing matrices as explained in Remark 1–1. We then train our *ReLU* network via Algorithm 1 to approximate $\mathbf{w}_*$, starting from the weight initialization $\mathbf{w}_1 = \mathbf{1} \in \mathbb{R}^r$ at the first iteration. At the t-th iteration of Algorithm 1, we draw iid input data samples $\mathbf{x}_{t_i}, i = 1, \ldots, b$, from a distribution fixed throughout the run, selected among the four aforementioned possible distributions. Given input data point $\mathbf{x}_{t_i}$, we instantiate a data-poisoning attack without explicitly checking for consistency with the assumptions of Theorem 1; we sample α_{t_i} from Bernoulli$(\beta(\mathbf{x}_{t_i}))$. Thus the probability of attack is $\beta(\mathbf{x}_{t_i}) = \Pr\left(\alpha_{\mathbf{x}_{t_i}} = 1\right)$, and if $\alpha_{t_i} = 1$, we set the additive distortion as, $\xi_{t_i} = \theta_* \mathbb{1}_{\{i(\mathrm{mod}\ 2)=0\}} - \theta_* \mathbb{1}_{\{i(\mathrm{mod}\ 2)\neq 0\}}$,where $\mathbb{1}_{\{\}}$ denotes the indicator function. We run SGD similarly to Algorithm 1.

Table 1 summarizes the configuration of hyperparameters across the eight simulation setups. When the noise bound θ_* varies, it takes values in $\{0, 0.125, 0.25, 0.5, 1, 2, 4\}$ and the probability β of adversarial attack is fixed to 0.5. When β varies, it takes values in $\{0.005, 0.05, 0.1, 0.2, 0.5, 0.9\}$ and θ_* is fixed to 0.25. Based on empirical tuning, the learning rate is set to $\eta = 0.0001$ when the input data distribution is $\mathcal{N}(\mu = 0, \sigma^2 = 1)$ or t($\nu = 4$), and to $\eta = 0.00005$ when the input data distribution is $\mathcal{N}(\mu = 0, \sigma^2 = 9)$ or Laplace($\mu = 0, b = 2$). The input data dimension is set to $n = 100$ for data sampled from $\mathcal{N}(\mu = 0, \sigma^2 = 1)$ or t($\nu = 4$), whereas it is set to $n = 50$ for data sampled from $\mathcal{N}(\mu = 0, \sigma^2 = 9)$ or Laplace($\mu = 0, b = 2$); smaller dimension n is used in the latter case due to higher variance in the input data, which can affect the numerical stability of Algorithm 1 and of SGD. The filter size, batch size and network width are set to $r = 25, b = 16$ and $k = 10$ across all setups.

Performance is measured in terms of the parameter recovery error $\|\mathbf{w}_t - \mathbf{w}_\star\|$ at iteration t of Algorithm 1 and of SGD, where $\|\cdot\|$ denotes the Euclidean norm. In all figures of this section and of Appendices F, G and H, the vertical and horizontal axes display recovery errors and iterations, respectively. Recovery error tick mark labels are shown in $\log_{10}$ scale, while the corresponding tick marks are shown in the original scale.

Fig. 1 provides a simulation-based validation of Theorem 1 regarding the performance of Algorithm 1 (Neuro-Tron). In each plot of Fig. 1, input data are sampled from a fixed distribution. On the left-hand side of Fig. 1, increasing the magnitude of attack (noise bound) $\theta_\star$ increases the parameter recovery error. On the right-hand side of Fig. 1, increasing the probability of attack β increases the parameter recovery error.

To further validate Neuro-Tron via simulation, Figure G.6 in Appendix G provides parameter recovery errors in the absence of data-poisoning attack ($\theta_\star = 0$). Recovery errors are in the vicinity of 10^{-14} for $\theta_\star = 0$ across different input data distributions, demonstrating the capacity of Neuro-Tron to recover network parameters under no attack. Moreover, Fig. G.6 shows an anticipated degradation in parameter recovery under relatively small magnitude of attack ($\theta_\star = 0.125$) when comparing to no attack ($\theta_\star = 0$).

Figs. 2, 3 in this section and Figures F.4, F.5 in Appendix F provide a simulation-based comparison between Neuro-Tron and SGD for different input data distributions, noise bounds $\theta_\star$ and probabilities β of attack. These figures provide empirical evidence that Neuro-Tron attains smaller parameter recovery error and faster rate of convergence than SGD under data-poisoning attacks.

We note that even with input data distributions, such as Laplace($\mu = 0, b = 2$) which have tails heavier than the Gaussian, we see in Fig. F.4 that Neuro-Tron retains its advantage over SGD. More strikingly, Neuro-Tron outperforms SGD under Student's t($\nu = 4$) distribution as seen in Fig. 3. Note that t($\nu = 4$) has infinite kurtosis (fourth moment), and therefore it is not covered by the assumptions of Theorem 1; nevertheless, our simulations demonstrate that Neuro-Tron attains analogous parameter recovery accuracy with t($\nu = 4$) as it does with the other three input data distributions.

5. Conclusion

In this paper, we provide the first provably robust training algorithm for a class of finite-width neural networks under a data-poisoning attack. In particular, we have constructed an iterative stochastic gradient-free algorithm which, up to a given level of parameter approximation accuracy and level of probabilistic confidence, performs supervised learning on a finite-width neural network in the presence of a malicious oracle adding noise to some true continuous output. We also establish that our performance guarantees are nearly-optimal in the worst case of attack on every output point.

We note that three immediate open questions arise based on the present results. Firstly, we recognize that in practice the question of defending against data-poisoning attacks becomes most relevant for deeper neural networks and for additional data distributions. Along these lines, it remains to extend our results to deeper neural networks and to data distributions with lesser number of moments being finite than assumed in Theorem 1. We anticipate two paths to be tractable; a preliminary step in this direction might be to build on the neural tangent kernel (NTK) regime analysis of stochastic gradient dynamics under data-poisoning attacks at depth-2 [52] and extend such results to the context of multilayer neural networks in the NTK regime. Secondly, if we move away from stochastic algorithms, then it might be possible to build on the results in Chatterjee [56] to control the gradient dynamics despite training data attacks, while not having to make any unrealistic assumptions of having asymptotically large widths as needed in the former path based on Wang et al. [52]. Secondly, an open question is to characterize the approximation accuracy and confidence trade-off of Theorem 1 as a function of the probability of adversarial attack. Thirdly, alternative adversarial attacks can be considered, conducting non-additive distortions to the output data or corrupting the input data.

CRediT authorship contribution statement

Sayar Karmakar: Methodology, Validation, Formal analysis, Writing - original draft, Writing - review & editing. **Anirbit Mukherjee:** Conceptualization, Methodology, Validation, Formal analysis, Writing - original draft, Writing - review & editing. **Theodore Papamarkou:** Methodology, Validation, Investigation, Writing - original draft, Writing - review & editing, Visualization.

Data availability

All data and code that have been used are linked to from the main file.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Anirbit Mukherjee reports financial support was provided by Johns Hopkins University. Anirbit Mukherjee reports financial support was provided by Adobe Research (San Jose). Anirbit Mukherjee reports financial support was provided by University of Pennsylvania Wharton School. Anirbit Mukherjee is a current associate editor of the Neurocomputing journal. Theodore Papamarkou has earlier been an associate editor at the Neurocomputing Journal.

Acknowledgements

We would like to thank Amitabh Basu for extensive discussions on various parts of this paper. The first author's research is supported by NSF DMS 2124222. The second author would like to thank the MINDS Data Science Fellowship of Johns Hopkins University for supporting this work. The second author would also like to acknowledge the extensive discussions on this topic with Anup Rao, Sridhar Mahadevan, Pan Xu and Wenlong Mou when he was interning at Adobe, San Jose, during the summer of 2019.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.neucom.2023.02.034.

References

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, arXiv preprint arXiv:1312.6199 (2013).
[2] I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, arXiv preprint arXiv:1412.6572 (2014).
[3] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z.B. Celik, A. Swami, Practical black-box attacks against machine learning, in: Proceedings of the 2017 ACM on Asia conference on computer and communications security, 2017, pp. 506–519.
[4] V. Behzadan, A. Munir, Vulnerability of deep reinforcement learning to policy induction attacks, in: International Conference on Machine Learning and Data Mining in Pattern Recognition, Springer, 2017, pp. 262–275.
[5] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, P. Abbeel, Adversarial attacks on neural network policies, arXiv preprint arXiv:1702.02284 (2017).
[6] N. Akhtar, A. Mian, Threat of adversarial attacks on deep learning in computer vision: A survey, IEEE Access 6 (2018) 14410–14430.
[7] S. Qiu, Q. Liu, S. Zhou, C. Wu, Review of artificial intelligence adversarial attack and defense technologies., Applied Sciences (2076–3417) 9 (2019).
[8] Z. Dou, S.J. Osher, B. Wang, Mathematical analysis of adversarial attacks, arXiv preprint arXiv:1811.06492 (2018).
[9] J. Lin, C. Song, K. He, L. Wang, J.E. Hopcroft, Nesterov accelerated gradient and scale invariance for improving transferability of adversarial examples, arXiv preprint arXiv:1908.06281 (2019).
[10] C. Song, K. He, L. Wang, J.E. Hopcroft, Improving the generalization of adversarial training with domain adaptation, arXiv preprint arXiv:1810.00740 (2018).
[11] S. Bubeck, E. Price, I. Razenshteyn, Adversarial examples from computational constraints, arXiv preprint arXiv:1805.10204 (2018).
[12] A. Degwekar, P. Nakkiran, V. Vaikuntanathan, Computational limitations in robust classification and win-win results, arXiv preprint arXiv:1902.01086 (2019).
[13] L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, A. Madry, Adversarially robust generalization requires more data, in: Advances in Neural Information Processing Systems, 2018, pp. 5014–5026.
[14] O. Montasser, S. Hanneke, N. Srebro, Vc classes are adversarially robustly learnable, but only improperly, arXiv preprint arXiv:1902.04217 (2019).
[15] A. Raghunathan, J. Steinhardt, P. Liang, Certified defenses against adversarial examples, arXiv preprint arXiv:1801.09344 (2018a).
[16] A. Raghunathan, J. Steinhardt, P.S. Liang, Semidefinite relaxations for certifying robustness to adversarial examples, in: Advances in Neural Information Processing Systems, 2018b, pp. 10877–10887.
[17] D. Yin, K. Ramchandran, P. Bartlett, Rademacher complexity for adversarially robust generalization, arXiv preprint arXiv:1810.11914 (2018).
[18] J. Khim, P.-L. Loh, Adversarial risk bounds via function transformation, arXiv preprint arXiv:1810.09519 (2018).
[19] Y. Wang, K. Chaudhuri, Data poisoning attacks against online learning, 2018. arXiv:1808.08994.
[20] X. Zhang, X. Zhu, L. Lessard, Online data poisoning attack, 2019. arXiv:1903.01666.
[21] P.W. Koh, J. Steinhardt, P. Liang, Stronger data poisoning attacks break data sanitization defenses, 2018. arXiv:1811.00741.
[22] A. Schwarzschild, M. Goldblum, A. Gupta, J.P. Dickerson, T. Goldstein, Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks, in: International Conference on Machine Learning, PMLR, 2021, pp. 9389–9398.
[23] F. Rosenblatt, The perceptron: a probabilistic model for information storage and organization in the brain, Psychological review 65 (1958) 386.
[24] S.K. Pal, S. Mitra, Multilayer perceptron, fuzzy sets, and classification, IEEE transactions on neural networks 3 (5) (1992) 683–697.
[25] Y. Freund, R.E. Schapire, Large margin classification using the perceptron algorithm, Machine learning 37 (1999) 277–296.
[26] S.M. Kakade, V. Kanade, O. Shamir, A. Kalai, Efficient learning of generalized linear and single index models with isotonic regression, in: Advances in Neural Information Processing Systems, 2011, pp. 927–935.
[27] A. Klivans, R. Meka, Learning graphical models using multiplicative weights, in: 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS), IEEE, 2017, pp. 343–354.
[28] S. Goel, A. Klivans, Learning depth-three neural networks in polynomial time, arXiv preprint arXiv:1709.06010 (2017).
[29] S. Goel, A. Klivans, R. Meka, Learning one convolutional layer with overlapping patches, arXiv preprint arXiv:1802.02547 (2018).
[30] T. Gu, B. Dolan-Gavitt, S. Garg, Badnets: Identifying vulnerabilities in the machine learning model supply chain, arXiv preprint arXiv:1708.06733 (2017).
[31] K. Liu, B. Dolan-Gavitt, S. Garg, Fine-pruning: Defending against backdooring attacks on deep neural networks, in: International Symposium on Research in Attacks, Intrusions, and Defenses, Springer, 2018, pp. 273–294.
[32] B. Tran, J. Li, A. Madry, Spectral signatures in backdoor attacks, arXiv preprint arXiv:1811.00636 (2018).
[33] A. Shafahi, W.R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, T. Goldstein, Poison frogs! targeted clean-label poisoning attacks on neural networks, arXiv preprint arXiv:1804.00792 (2018).
[34] C. Zhu, W.R. Huang, H. Li, G. Taylor, C. Studer, T. Goldstein, Transferable clean-label poisoning attacks on deep neural nets, in: International Conference on Machine Learning, PMLR, 2019, pp. 7614–7623.
[35] B. Biggio, B. Nelson, P. Laskov, Support vector machines under adversarial label noise, in: Asian conference on machine learning, PMLR, 2011, pp. 97–112.
[36] H. Xiao, B. Biggio, G. Brown, G. Fumera, C. Eckert, F. Roli, Is feature selection secure against training data poisoning?, 2018. arXiv:1804.07933.
[37] M. Zhao, B. An, W. Gao, T. Zhang, Efficient label contamination attacks against black-box learning models, in: IJCAI, 2017, pp. 3945–3951.
[38] I. Diakonikolas, T. Gouleakis, C. Tzamos, Distribution-independent pac learning of halfspaces with massart noise, arXiv preprint arXiv:1906.10075 (2019).
[39] J. Cohen, E. Rosenfeld, Z. Kolter, Certified adversarial robustness via randomized smoothing, in: International Conference on Machine Learning, PMLR, 2019, pp. 1310–1320.
[40] E. Rosenfeld, E. Winston, P. Ravikumar, Z. Kolter, Certified robustness to label-flipping attacks via randomized smoothing, in: International Conference on Machine Learning, PMLR, 2020, pp. 8230–8241.
[41] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, B. Li, Manipulating machine learning: Poisoning attacks and countermeasures for regression learning, 2018. arXiv:1804.00308.
[42] H. Xu, C. Caramanis, S. Mannor, Robust regression and lasso, 2008. arXiv:0811.1790.
[43] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, Y. Bengio, Show, attend and tell: Neural image caption generation with visual attention, in: International Conference on Machine Learning, 2015, pp. 2048–2057.
[44] J. Feng, H. Xu, S. Mannor, S. Yan, Robust logistic regression and classification, in: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1, NIPS'14, MIT Press, Cambridge, MA, USA, 2014, p. 253–261.
[45] Y. Chen, C. Caramanis, S. Mannor, Robust sparse regression under adversarial corruption, in: Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28, ICML'13, JMLR.org, 2013, p. III–774–III–782.
[46] C. Liu, B. Li, Y. Vorobeychik, A. Oprea, Robust linear regression against training data poisoning, in: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec '17, Association for Computing Machinery, New York, NY, USA, 2017, p. 91–102.
[47] X. Li, Compressed sensing and matrix completion with constant proportion of corruptions, 2011. arXiv:1104.1041.
[48] J.N. Laska, M.A. Davenport, R.G. Baraniuk, Exact signal recovery from sparsely corrupted measurements through the pursuit of justice, in: 2009 Conference Record of the Forty-Third Asilomar Conference on Signals, Systems and Computers, 2009, pp. 1556–1560.
[49] N.H. Nguyen, T.D. Tran, Exact recoverability from dense corrupted observations via l_1 minimization, 2011. arXiv:1102.1227.

[50] R. Gao, T. Cai, H. Li, C.-J. Hsieh, L. Wang, J.D. Lee, Convergence of adversarial training in overparametrized neural networks, in: Advances in Neural Information Processing Systems, 2019, pp. 13009–13020.
[51] M. Li, M. Soltanolkotabi, S. Oymak, Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 4313–4324.
[52] Y. Wang, P. Mianjy, R. Arora, Robust learning for data poisoning attacks, in: M. Meila, T. Zhang (Eds.), Proceedings of the 38th International Conference on Machine Learning, volume 139 of Proceedings of Machine Learning Research, PMLR, 2021, pp. 10859–10869.
[53] J. Steinhardt, P.W.W. Koh, P.S. Liang, Certified defenses for data poisoning attacks, in: Advances in neural information processing systems, 2017, pp. 3517–3529.
[54] M. Barreno, B. Nelson, A.D. Joseph, J.D. Tygar, The security of machine learning, Machine Learning 81 (2010) 121–148.
[55] S. Karmakar, A. Mukherjee, Provable training of a relu gate with an iterative non-gradient algorithm, Neural Networks 151 (2022) 264–275.
[56] S. Chatterjee, Convergence of gradient descent for deep neural networks, 2022. https://arxiv.org/abs/2203.16462. 10.48550/ARXIV.2203.16462.

Sayar obtained his bachelors (B.Stat hons.) and masters (M.Stat) degrees at Indian Statistical Institute, Kolkata and went on to pursue a PhD in Statistics at the University of Chicago. After graduation, he joined the Statistics department at University of Florida as an Assistant Professor. His main research interests lie in exploring different forms of dependence present in the data. Over the past few years, he is working on different projects in multiple time series, time-varying models, econometrics and deep learning theory. He has served as a referee for leading journals in Statistics and Econometrics and was recently appointed as an Early Career Advisory Board member at the Journal of Multivariate Analysis. His research is partially supported by NSF, AMS Simons and UF Informatics Institute.

Anirbit obtained his undergraduate and master's degrees in physics at the Chennai Mathematical Institute (CMI) and at the Tata Institute of Fundamental Research (TIFR), respectively. In 2020, he completed his Ph.D. in the department of applied mathematics and statistics at Johns Hopkins University (JHU). His doctoral research was recognized by two fellowships from JHU, "MINDS Data Science Fellowship" and the "Walter L. Robb Fellowship". Anirbit is currently an assistant professor at The University of Manchester and for a brief period prior to that, he was a post-doc at the Wharton, the Department of Statistics at the University of Pennsylvania. Anirbit specializes in provable training algorithms and generalization guarantees for neural networks. Most recently, Anirbit has been selected as a member of the European Laboratory for Learning and Intelligent Systems (ELLIS) Society.

Theodore Papamarkou works as a reader in the mathematics of data science at The University of Manchester. Prior to his current role, Theodore held the positions of strategic hire in artificial intelligence at the Oak Ridge National Laboratory, and of assistant professor at the University of Glasgow. In early stages of his career, he worked as a post-doctoral researcher at the University of Warwick, at University College London, and at the University of Cambridge. Theodore's research spans probabilistic machine learning. Theodore is interested in addressing questions related to uncertainty quantification and to approximate inference with big data or with high-dimensional models. Theodore is a member of the ELLIS society.