International Journal of Machine Learning and Cybernetics
https://doi.org/10.1007/s13042-024-02363-5

ORIGINAL ARTICLE

Backdoor learning curves: explaining backdoor poisoning beyond influence functions

Antonio Emanuele Cinà[1] · Kathrin Grosse[3] · Sebastiano Vascon[2] · Ambra Demontis[4] · Battista Biggio[4] · Fabio Roli[1] · Marcello Pelillo[2]

Received: 17 July 2023 / Accepted: 6 August 2024
© The Author(s) 2024

Abstract
Backdoor attacks inject poisoning samples during training, with the goal of forcing a machine learning model to output an attacker-chosen class when presented with a specific trigger at test time. Although backdoor attacks have been demonstrated in a variety of settings and against different models, the factors affecting their effectiveness are still not well understood. In this work, we provide a unifying framework to study the process of backdoor learning under the lens of incremental learning and influence functions. We show that the effectiveness of backdoor attacks depends on (i) the complexity of the learning algorithm, controlled by its hyperparameters; (ii) the fraction of backdoor samples injected into the training set; and (iii) the size and visibility of the backdoor trigger. These factors affect how fast a model learns to correlate the presence of the backdoor trigger with the target class. Our analysis unveils the intriguing existence of a region in the hyperparameter space in which the accuracy of clean test samples is still high while backdoor attacks are ineffective, thereby suggesting novel criteria to improve existing defenses.

Keywords Backdoor poisoning · Influence functions · Poisoning · Machine learning · Adversarial machine learning · Security

1 Introduction

Machine learning models are vulnerable to backdoor poisoning [1–4]. These attacks consist of injecting poisoning samples at training time, with the goal of forcing the trained model to output an attacker-chosen class when presented with a specific trigger at test time, while working as expected otherwise. To this end, the poisoning samples typically need not only to embed such a *backdoor trigger* themselves, but also to be labeled as the attacker-chosen class. As backdoor poisoning preserves model performance on clean test data, it is not straightforward for the victim to realize that the model has been compromised.

Backdoor poisoning has been demonstrated in a plethora of scenarios [3–5]. In the most common scenario, the user is assumed to download a pre-trained, backdoored model from an untrusted source, to subsequently fine-tune it on their data [3]. As backdoors typically remain effective even after this fine-tuning step, they may successfully be exploited by the attacker at test time [1, 2]. Alternatively, the attacker is assumed to alter part of the training data collected by the user, either to train the model from scratch or to fine-tune a

✉ Antonio Emanuele Cinà
antonio.cina@unige.it

Kathrin Grosse
kathrin.grosse@epfl.ch

Sebastiano Vascon
sebastiano.vascon@unive.it

Ambra Demontis
ambra.demontis@unica.it

Battista Biggio
battista.biggio@unica.it

Fabio Roli
fabio.roli@unige.it

Marcello Pelillo
pelillo@unive.it

1 DIBRIS, University of Genoa, Genoa, Italy

2 DAIS, Ca' Foscari University of Venice, Venezia, Italy

3 VITA Lab, École Polytechnique Fédéerale de Lausanne, Lausanne, Switzerland

4 DIEE, University of Cagliari, Cagliari, Italy

Published online: 26 September 2024

pre-trained model via transfer learning [6, 7]. In such cases, the training labels of poisoning samples may be either fully controlled by the attacker [6] or subject to validation by the victim [7], depending on the considered threat model. Backdoor attacks have been shown to be effective in different applications and against a variety of models, including vision [1] and language models [8], graph neural networks [9], and even reinforcement learning [10]. However, it is still unclear which factors influence the ability of a model to learn a backdoor, i.e., to classify the test samples containing the backdoor trigger as the attacker-chosen class.

In this work, we analyze the process of *backdoor learning* to identify the main factors affecting the vulnerability of machine learning models against this attack. To this end, we propose a framework that characterizes the backdoor learning process. The learning process of humans is usually portrayed using learning curves, a graphical representation of the relationship between the hours spent practicing and the proficiency to accomplish a given task. Inspired by this concept, we introduce the notion of *backdoor learning curves* (Sect. 2). To generate these curves, we formulate backdoor learning as an incremental learning problem and assess how the loss on the backdoor samples decreases as the target model gradually learns them. These backdoor learning curves are independent of the threat model as they capture training data and fine-tuning attacks. The slope of this curve, the *backdoor learning slope*, which is connected to the notion of influence functions, quantifies the speed with which the model learns the backdoor samples and hence its vulnerability. Additionally, to provide further insights about the backdoor's influence on the learned classifiers, we propose a way to quantify the *backdoor impact on learning parameters*, i.e., how much the parameters of a model deviate from the initial values when the model learns a backdoor.

Our experimental analysis (Sect. 3) shows that the factors influencing the success of backdoor poisoning are: (i) the fraction of backdoor samples injected into the training data; (ii) the size of the backdoor trigger; and (iii) the complexity of the target model, controlled via its hyperparameters. Concerning the latter, our experimental findings reveal a region in the hyperparameter space where models are highly accurate on clean samples while also being robust to backdoor poisoning. This region exists as, to learn a backdoor, the target model is required to increase the complexity of its decision function significantly, and this is not possible if the model is sufficiently regularized. This observation will help identify novel criteria to improve existing defenses and inspire new countermeasures.

To summarize, the main contributions of this work are:

- We introduce *backdoor learning curves* as a powerful tool to thoroughly characterize the backdoor learning process;
- We introduce a metric, named *backdoor learning slope*, to quantify the ability of the classifier to learn backdoors;
- We identify three important factors that affect the vulnerability against backdoors;
- We unveil a region in the hyperparameter space in which the classifiers are highly accurate and robust against backdoors, which supports novel defensive strategies.

We conclude the paper by discussing related work in Sect. 4, along with the limitations of our approach and promising future research directions in Sect. 5.

2 Backdoor learning curves

In this section, we introduce our framework to characterize backdoor poisoning by means of learning curves and their slope. Afterward, we introduce two measures to quantify the backdoor impact on the model's parameters.

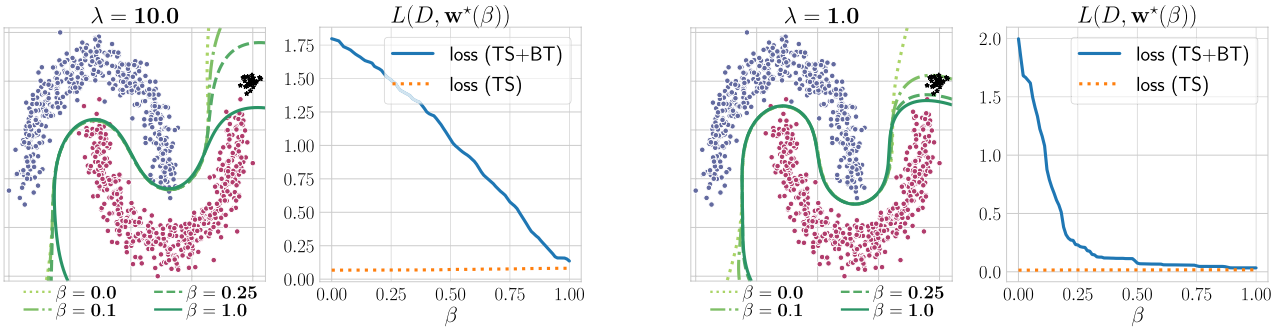
Notation. We denote the input data and their labels respectively with $\boldsymbol{x} \in \mathbb{R}^d$ and $y \in \{1, .., c\}$, being c the number of classes. We refer to the untainted, clean training data as $\mathcal{D}_{\text{tr}} = (\boldsymbol{x}_i, y_i)_{i=1}^n$, and to the backdoor samples injected into the training set as $\mathcal{P}_{\text{tr}} = (\hat{\boldsymbol{x}}_j, \hat{y}_j)_{j=1}^m$. We refer to the clean test samples as $\mathcal{D}_{\text{ts}} = (\boldsymbol{x}_t, y_t)_{t=1}^k$ and to the test samples containing the backdoor trigger as $\mathcal{P}_{\text{ts}} = (\hat{\boldsymbol{x}}_t, \hat{y}_t)_{t=1}^k$.

Backdoor learning curves. We leverage previous work from incremental learning [11, 12] to study how gradually incorporating backdoor samples affects the learned classifier. In mathematical terms, the learning problem can be formalized as:

$$\begin{aligned} \boldsymbol{w}^\star(\beta) &\in \arg\min_{\boldsymbol{w}} \mathcal{L}(\mathcal{D}_{\text{tr}} \cup \mathcal{P}_{\text{tr}}, \boldsymbol{w}) \\ &= L(\mathcal{D}_{\text{tr}}, \boldsymbol{w}) + \beta L(\mathcal{P}_{\text{tr}}, \boldsymbol{w}) + \lambda \Omega(\boldsymbol{w}), \end{aligned} \tag{1}$$

where L is the loss attained on a given dataset by the classifier with parameters $\boldsymbol{w}$, and $\mathcal{L}$ is the loss computed on the training points and the backdoor samples, which also includes a regularization term $\Omega(\boldsymbol{w})$ (e.g., $\|\boldsymbol{w}\|_2^2$), weighed by the regularization hyperparameter λ. To gradually incorporate the backdoor samples $\mathcal{P}_{\text{tr}}$ into the learning process, we introduce the hyperparameter $\beta \in [0, 1]$, and increase it from 0 (unpoisoned classifier) to 1 (poisoned classifier). As β increases, the classifier gradually learns the backdoor by adjusting its parameters; for this reason, we make the dependency of the optimal weights $\boldsymbol{w}^\star$ on β explicit as $\boldsymbol{w}^\star(\beta)$.[1]

[1] Recall that the formulation reported in Eq. (1) encompasses many existing learning algorithms, including support vector machines (SVMs), ridge and logistic classifiers. For example, considering either $\beta = 0$ or $\beta = 1$, the SVM learning problem amounts to minimizing $C \cdot \big(L(\mathcal{D}_{\text{tr}}, \boldsymbol{w}) + \beta L(\mathcal{P}_{\text{tr}}, \boldsymbol{w})\big) + \frac{1}{2}\|\boldsymbol{w}\|_2^2$, which is equivalent to



(a) Strong regularization, $\lambda = 10$ ($C = 0.1$). (b) Weak regularization, $\lambda = 1$ ($C = 1$).

Fig. 1 Backdoor learning curves. Considering an SVM with the RBF kernel ($\gamma = 10$) on a toy dataset in two dimensions, we show the influence of model complexity (controlled by the regularization hyperparameter $\lambda = \frac{1}{C}$) on backdoor learning. For both the strong (*left*) and weak (*right*) regularization settings, we report two plots. The left plot shows the two-dimensional data (dots) and decision surface for different values of β (green lines). The right plot shows the backdoor learning curve, i.e. how the loss decreases as β ranges from 0 to 1, which amounts to learning the backdoor samples. We plot both the loss on the clean test samples (orange dotted line) and on the test samples with the backdoor trigger (blue line). The slope of these curves represents the speed with which the model learns to classify the backdoor samples (black dots) as blue dots, unveiling that strong regularization slows down such a process

We now define the *backdoor learning curve* as the curve showing the behavior of the classifier loss $L(\mathcal{P}_{\text{ts}}, \boldsymbol{w}^\star(\beta))$ on the test samples with the backdoor trigger as a function of β. In the following, we abbreviate $L(\mathcal{P}_{\text{ts}}, \boldsymbol{w}^\star(\beta))$ as L. Intuitively, the faster the backdoor learning curve decreases, the easier the target model is backdoored. The exact details of how the model is backdoored do not matter for this analysis, e.g. our approach captures for example both the setting where the training data is altered as well as the setting where fine-tuning data is tampered with.

We give an example of two such curves under different regularizations in Fig. 1. The left plots depict a strongly regularized classifier. The corresponding backdoor learning curve (on the right) shows that the classifier achieves low loss and high accuracy on the backdoor samples only after poisoning (when $\beta = 1$), i.e. even when the loss on the backdoor samples is considered equally important to the loss on the training samples. The classifier on the right, instead, is less regularized and thus more complex. Consequently, this classifier learns to incorporate the backdoor samples much faster (at low β), namely when the loss on the backdoor points is taken into account less than the one on the training data. This highlights that this classifier is probably more vulnerable to this attack.

Backdoor learning slope. We quantify how fast an untainted classifier can be poisoned by proposing a novel measure, namely the *backdoor learning slope*, that measures the velocity with which the classifier learns to classify the backdoor samples correctly. This measure allows us to compare the vulnerability of a classifier trained with different hyperparameters or consider different poisoning scenarios (e.g. when the attacker can inject a different number of poisoning points or create triggers with different sizes), allowing us to identify factors relevant to backdoor learning. Moreover, as we will show, this measure can be used by the system designer to choose an appropriate combination of hyperparameters for the task at hand. To this end, we define the *backdoor learning slope* as the gradient of the backdoor learning curve at $\beta = 0$, capturing the velocity of the curve on learning the backdoor. Formally, the *backdoor learning slope* can be formulated as follow:

$$\frac{\partial L(\mathcal{P}_{\text{ts}}, \boldsymbol{w}^\star(\beta))}{\partial \beta} = \frac{\partial L}{\partial \boldsymbol{w}} \frac{\partial \boldsymbol{w}^\star}{\partial \beta}, \tag{2}$$

where the first term is straightforward to compute, and the second term implicitly captures the dependency of the optimal weights on the hyperparameter β. In other words, it requires us to understand how the optimal classifier parameters change when gradually increasing β from 0 to 1, i.e., while incorporating the backdoor samples into the learning process.

To compute this term, as suggested in previous work in incremental learning [11], we assume that, while increasing β, the solution maintains the optimality (Karush–Kuhn–Tucker, KKT) conditions intact. This equilibrium implies that $\nabla_\beta \nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}^\star) + \frac{\partial \boldsymbol{w}^\star}{\partial \beta} \nabla^2_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}^\star) = \boldsymbol{0}$. Based on this condition, we obtain the derivative of interest,

Footnote 1 (Continued)

our formulation if one sets L to be the hinge loss, $\Omega(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|_2^2$, and $\lambda = \frac{1}{C}$.

$$\frac{\partial \boldsymbol{w}^\star}{\partial \beta} = -(\nabla^2_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}^\star))^{-1} \cdot \nabla_\beta \nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}^\star). \tag{3}$$

Substituting it in Eq. 2 we obtain the complete gradient:

$$\frac{\partial L(\mathcal{P}_{\text{ts}}, \boldsymbol{w}^\star(\beta))}{\partial \beta} = -\nabla_{\boldsymbol{w}} L \cdot (\nabla^2_{\boldsymbol{w}} \mathcal{L})^{-1} \cdot \nabla_\beta \nabla_{\boldsymbol{w}} \mathcal{L}. \tag{4}$$

The gradient in Eq. 4 corresponds to the sum of the pairwise influence function values $\mathcal{I}_{\text{up, loss}}(\boldsymbol{x}_{\text{tr}}, \boldsymbol{x}_{\text{ts}})$ used by Koh et al. [13]. The authors indeed proposed to measure how relevant a training point is for the predictions of a given test point by computing $\frac{\partial L}{\partial \beta}\Big|_{\beta=0} = \sum_t \sum_j \mathcal{I}_{\text{up, loss}}(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{x}}_j)$. To understand how this gradient can be efficiently computed via Hessian-vector products and other approximations, we refer the reader to [13] as well as to recent developments in gradient-based bilevel optimization [14–16]. Moreover, we show in Sect. 3.2.5 (Figs. 13 and 14) an example of the usage of influence functions for weakly and strongly regularized models.

The main difference between the approach by Koh et al. [13] and ours stems from their implicit treatment of regularization and our interest in understanding vulnerability to a subset of backdoor training points, rather than in providing prototype-based explanations. However, directly using the gradient of the loss wrt. β comes with two disadvantages. First, the slope is inverse to β, and second, to obtain results comparable across classifiers, we need to rescale the slope. We thus transform the gradient as:

$$\theta = -\frac{2}{\pi} \arctan\left(\frac{\partial L}{\partial \beta}\Big|_{\beta=0} \right) \in [-1, 1], \tag{5}$$

where we use the negative sign to have positive values correlated with faster backdoor learning (i.e., the loss decreases faster as β grows). Computing $2/\pi$ of the gradient allows us to rescale the slope to be in the interval between $[-1, 1]$. Hence, a value around 0 implies that the loss of the backdoor samples does not decrease. In other words, the classifier does not learn the backdoor trigger and is hence very robust.

Backdoor impact on learning parameters. After introducing the previous plot and measure, we can quantify how backdoors are learned by the model. To provide further insights about the backdoor's influence on the learned classifier, we propose to monitor how the classifier's parameters deviate from their initial, unbackdoored values once a backdoor is added. Our approach below captures only convex learners. As shown by Zhang et al. [17], the impact of a network weight in non-convex classifiers' decisions depends on the layer of which it is part. Therefore, measuring the parameter deviation in the non-convex case is challenging, and we leave this unsolved problem for future work.

To capture the backdoor impact on learning parameters in the convex case, we consider the initial weights $\boldsymbol{w}_0 = \boldsymbol{w}^\star(\beta = 0)$ and $\boldsymbol{w}_\beta = \boldsymbol{w}^\star(\beta)$ for $\beta > 0$, and measure two quantities:

$$\rho = \|\boldsymbol{w}_\beta\| \in [0, \infty), \text{ and } \nu = \frac{1}{2}\left(1 - \frac{\boldsymbol{w}_0^\top \boldsymbol{w}_\beta}{\|\boldsymbol{w}_0\| \|\boldsymbol{w}_\beta\|}\right) \in [0, 1]. \tag{6}$$

The first measure, ρ, quantifies the change of the weights when β increases. This quantity is equivalent to the regularization term used for learning. The second one, ν, quantifies the change in orientation of the classifier. In a nutshell, we compute the angle between the two vectors and rescale it to be in the interval of [0, 1]. Both metrics are defined to grow as $\beta \to 1$, in other words the backdoored classifier deviates more and more from the original classifier. Consequently, in the empirical parameter deviation plots in Sect. 3.2, we report the value of $\rho(\nu)$ (on the y-axis) as β (on the x-axis) varies from 0 to 1, to show how the classifier parameters are affected by backdoor learning.

3 Experiments

Employing the previously proposed methodology, we carried out an empirical analysis of linear and nonlinear classifiers. In this section, we start with the experiments aimed at studying the impact of different factors on backdoor learning. To this end, we employ the backdoor learning curves and the backdoor learning slope to study how the capacity of the model to learn backdoors changes when (a) varying the model's complexity, defined by its hyperparameters, (b) the attacker's strength, defined by the percentage of poisoning samples in the training set and (c) the trigger size and visibility. Our results show that these components significantly determine how fast the backdoor is learned, and consequently, the model's vulnerability. Then, leveraging the proposed measures to analyze how the classifier's parameters change during backdoor learning, we provide further insights into the effect of the aforementioned factors on the trained model. The results presented in this section will help identify novel criteria to improve existing defenses and inspire new countermeasures. The source code is available on the author's GitHub page.[2]

3.1 Experimental setup

Our work investigates which factors influence backdoor vulnerability considering convex learners and neural networks.

[2] https://github.com/Cinofix/backdoor_learning_curves.

In the following, we describe our datasets, models, and the backdoor attacks studied in our experiments.

Datasets. We carried out our experiments on MNIST [18], CIFAR10 [19] and Imagenette [20].[3] Supplementary details on the datasets are reported in Appendix A.

When adopting convex learners, we consider the two-class subproblems as in the work by Saha et al. [7] and Suya et al. [21]. On MNIST, we choose the pairs 7 vs 1, 3 vs 0, and 5 vs 2, as our models exhibited the highest clean accuracy on these pairs. On CIFAR10, analogous to prior work [7], we choose *airplane vs frog*, *bird vs dog*, and *airplane vs truck*. On Imagenette we randomly choose *tench vs truck*, *cassette player vs church*, and *tench vs parachute*. For each two-class subtask, we use 1500 and 500 samples as training and test set respectively. In the following section, we focus on the results of one pair on each dataset: 7 vs 1 on MNIST, *airplane vs frog* on CIFAR10, and *tench vs truck* on Imagenette. The results of the other pairs (reported in Appendix B) are analogous. When testing our framework against neural networks, we train on all ten classes of Imagenette. We use 70% and 30% of the entire dataset for training and testing, respectively.

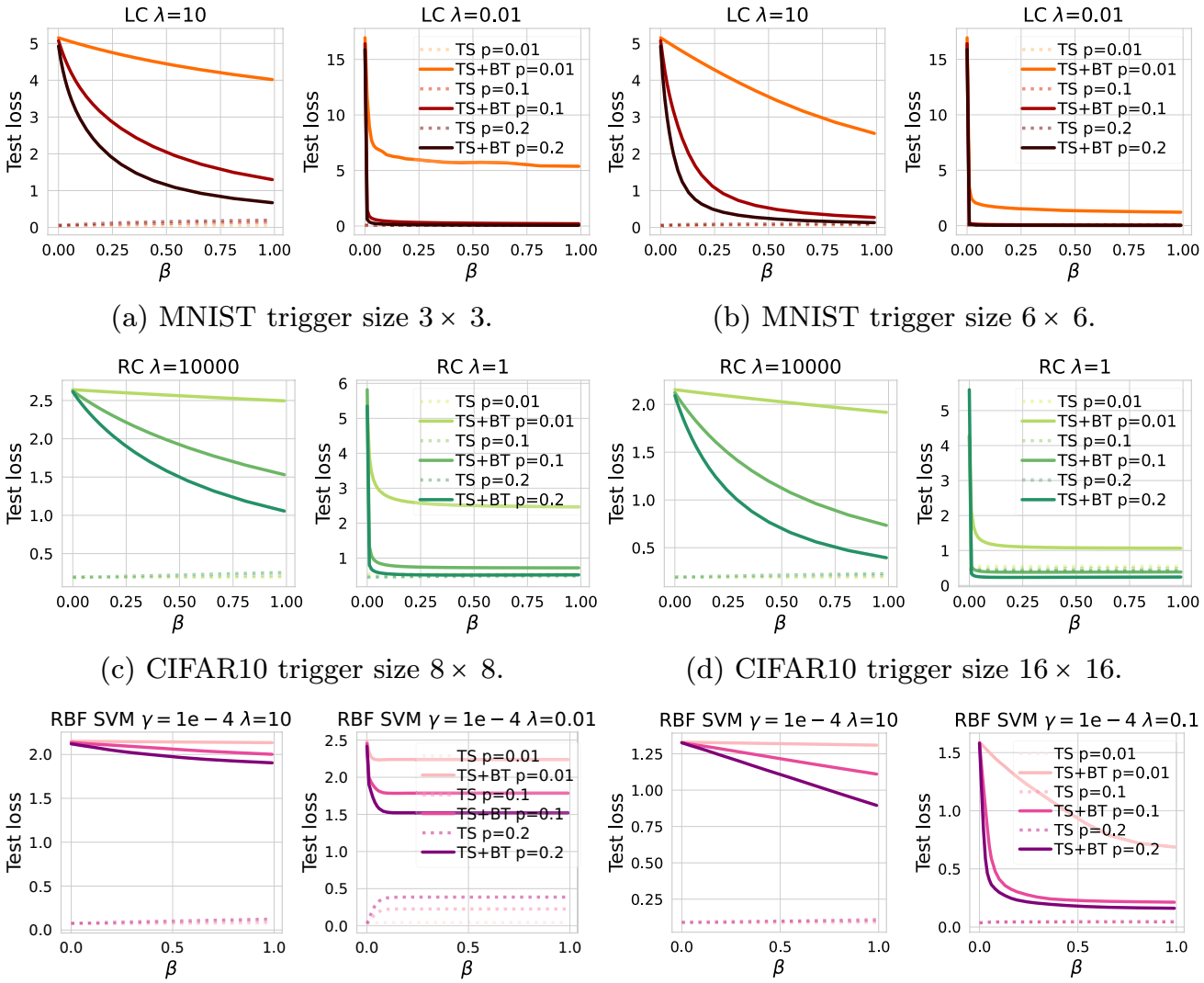
Models and training phase. To thoroughly analyze how learning a backdoor affects a model, we consider different convex learning algorithms, including linear Support Vector Machines (SVMs), Logistic Regression Classifiers (LCs), Ridge Classifiers (RCs), nonlinear SVMs using the Radial Basis Function (RBF) kernel, and deep neural networks. We train the classifiers directly on the pixel values scaled in the range [0, 1] on the MNIST dataset. For CIFAR10 and Imagenette, we instead consider a transfer learning setting frequently adopted in the literature [13, 22, 23]. Like Saha et al. [7], we use the pre-trained model AlexNet [24] as a feature extractor. The convex learners are then trained on top of the feature extractor. We study these convex learners due to their broad usage in industry [21], derived from their excellent results with smaller dataset [25], and good interpretability [26, 27]. In addition, we include in our evaluation pre-trained Resnet18 and Resnet50 [28] deep neural networks, sourced from Torchvision [29], which stand as some of the most extensively employed architectures [17]. These networks are fine-tuned to classify samples from the Imagenette dataset accurately.

Hyperparameters. The choice of hyperparameters has a relevant impact on the learned decision function. For example, some of these parameters control the complexity of the learned function, which may lead to overfitting [30], thereby potentially compromising classification accuracy on test samples. We argue that a high complexity may also lead to higher importance to outlying samples, including backdoors, and thus has a crucial impact on the capacity of the model to learn backdoors. To verify our hypothesis, we consider different configurations of the models' hyperparameters. For convex learners, we study two hyperparameters that impact model complexity, i.e., the regularization hyperparameter $\lambda = \frac{1}{C}$ and the RBF kernel hyperparameter γ. To this end, we take 10 values for λ on a uniformly spaced interval on a log scale from 1e−04 to 1e+02. For the Imagenette dataset we extend this interval in [1e−05, 1e+02]. Concerning the RBF kernel, we let γ take 5 uniformly spaced values on a log scale in [5e−04, 5e−02] for MNIST, [1e−04, 1e−02] for CIFAR, and [1e−05, 1e−03] for Imagenette. Furthermore, we take 10 values of λ in the log scale uniformly spaced interval [1e−01, 1e+02] for the RBF kernel. This allowed us to study a combination of 10 and 50 hyperparameters for linear classifiers and RBF SVM, respectively.

For deep neural networks, we consider two different numbers of epochs: 10 and 50, and increase the number of neurons when using Resnet50 instead of Resnet18. Whereas size intuitively correlates with complexity, previous works, including [31], show that decreasing the number of training epochs reduces the complexity of the trained network as well. Conversely, increasing epochs leads to overfitting on the training dataset, thus, a more complex decision function. Each network is fine-tuned using the SGD optimizer with a learning rate of 0.001, a momentum of 0.9, and a batch size of 256.

Backdoor attacks. We implement the backdoor attacks proposed by Gu et al. [1] against MNIST and CIFAR10. More concretely, we use a random 3×3 patch as the trigger for MNIST, while on CIFAR10, we increase the size to 8×8 to strengthen the attack [7]. We add the trigger pattern in the lower right corner of the image [1]. Samples from MNIST and CIFAR10 with and without trigger can be found in Fig. 12. However, in contrast to previous approaches [1], we use a separate trigger for each base-class i. The reason is that our study encompasses linear models that are unable to associate the same trigger pattern to two different classes. Using different trigger patterns, we enhance the effectiveness of the attack on these linear models. On the Imagenette dataset, we use the backdoor trigger developed by [32]. This attack injects a patterned perturbation mask into training samples to open the backdoor. A constant value c_m refers to the maximum allowed intensity. We apply the backdoor attacks to 10% of the training data if not stated otherwise, and, as done by Gu et al. [1], we force the backdoored model to predict the i-th class as class $(i + 1)\%n\_classes$ when the trigger is shown. We also report additional experiments concerning variations in the trigger's size or visibility.

[3] Imagenette is a subset of 10 classes from Imagenet. We use the 320 px version, where the shortest side of each image is resized to that size.



(a) MNIST trigger size 3×3.

(b) MNIST trigger size 6×6.

(c) CIFAR10 trigger size 8×8.

(d) CIFAR10 trigger size 16×16.

(e) Imagenette trigger visibility $c_m = 10$.

(f) Imagenette trigger visibility $c_m = 75$.

Fig. 2 Backdoor learning curves for: (top row) logistic classifier (LC) on MNIST 7 vs. 1 with $\lambda \in \{10, 0.01\}$ and trigger size 3×3 *(left)* or 6×6 (right); (middle row) Ridge classifier on CIFAR10 *airplane vs frog* with $\lambda \in \{100000, 100\}$ and trigger size 8×8 (left) or 16×16 (right); (bottom row) RBF SVM with $\gamma = 1e{-}04$ on Imagenette *tench vs truck* with $\lambda \in \{10, 0.1\}$ and trigger visibility $c_m = 10$ (left) or $c_m = 75$ (right). Darker lines represent a higher fraction of poisoning samples p injected into the training set. We report the loss on the clean test samples (TS) with a dashed line and on the test samples with the backdoor trigger (TS+BT) with a solid line

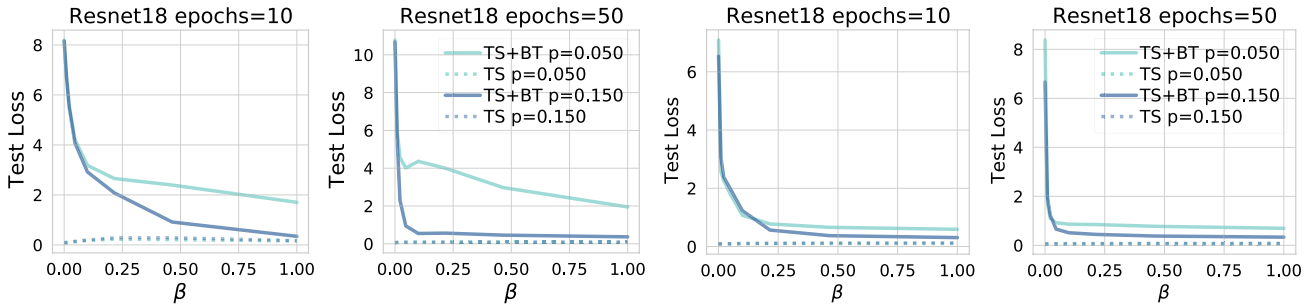
3.2 Experimental results

In the following, we now discuss our experimental results obtained with the datasets, classifiers, and backdoor attacks described above.

3.2.1 Backdoor learning curves

Here we present the results obtained using the learning curves that we proposed to study the impact of three different factors on the backdoor learning process: (i) *model complexity*, (ii) the *fraction of backdoor samples injected*, and (iii) the *size and visibility of the backdoor trigger*. We report the impact of these factors on the backdoor learning curves in Figs. 2 and 3. More specifically, in Fig. 2 we consider convex classifiers (i.e. LC, RC and RBF SVM) trained on two-class subproblems (MNIST, CIFAR10, and Imagenette), whereas in Fig. 3 we show the results for Resnet18 trained on all the ten classes of Imagenette.

To analyze the first factor, we report the results on the same classifiers, changing the hyperparameters that influence their corresponding complexity. In the case of convex learners, we test different values of the regularization coefficient, while for Resnet18, we increase the number of epochs.



(a) Imagenette with trigger visibility $c_m = 10$. (b) Imagenette with trigger visibility $c_m = 75$.

Fig. 3 Backdoor learning curves for Resnet18 trained on the full Imagenette training dataset with 10 and 50 epochs. Darker lines represent a higher fraction of poisoning samples p injected into the training set. We report the loss on the clean test samples (TS) with a dashed line and on the test samples with the backdoor trigger (TS+BT) with a solid line

To analyze the impact of the second factor, we plot the backdoor learning curves when the attacker injects an increasing percentage of poisoning points $p \in \{0.01, 0.1, 0.2\}$ for convex learners and $p \in \{0.05, 0.15\}$ for Resnet18. Finally, to study the third factor, namely the size and visibility of the backdoor trigger, we have created the same backdoor curves doubling the size of the patch triggers for MNIST and CIFAR10, and increasing the trigger's visibility for Imagenette. Even when a high percentage of poisoning points are injected, for flexible enough classifiers, the loss on the clean test samples remains almost constant. Instead, the loss on the test set containing the backdoor trigger is highly affected by the factors mentioned above. Both a smaller λ or a larger number of epochs (low regularization and thus higher complexity), and larger p (a high percentage of poisoning points added) increase the slope of the backdoor learning curve. This means that the classifier learns the backdoor faster. When the classifier is sufficiently complex, even a low percentage of low-poisoning points is enough to rapidly induce the classifier to learn the backdoor. On the other hand, this does not apply to highly regularized classifiers, which generally exhibit slower learning of backdoors. Therefore, limiting the classifier's complexity by selecting an appropriate regularization coefficient may mitigate vulnerability to backdoors. Furthermore, our results demonstrate that larger trigger sizes lead to faster backdoor learning by classifiers, particularly when they are not regularized. This observation holds when increasing the trigger's visibility, highlighting the well-known trade-off between the attacker's strength and detectability as introduced by Frederickson et al. [33]. The attacker can enhance the backdoor's effectiveness by increasing the trigger size or its visibility. However, these adjustments also make it easier for the defender to detect the attack.

Concerning the RBF SVM's robustness to backdoors, we analyzed the backdoors' learning curves for different values of γ, which determine the RBF kernel's curvature. More precisely, depending on γ, we have analyzed the backdoor learning curves, and the classifier's parameters change due to backdoor learning. We depict the learning curves in Fig. 4. On both datasets, reducing γ results in flatter backdoor learning curves and increased test loss, indicating greater robustness.

Remarks. Overall, our experiments show that to learn a backdoor, a classifier has to increase its complexity (if it is not already highly complex). However, an increase in complexity is limited when the classifier is highly regularized or when the attack strength is constrained. For this reason, highly regularized classifiers are preferable in terms of backdoor robustness. We present supplementary results for other classifiers in Appendix B, validating the trends above desc ribed.

3.2.2 Backdoor slope

From the previous results, we have seen that reducing complexity through regularization increases robustness against backdoors. For a deeper understanding of model complexity on backdoor learning, we leverage the proposed backdoor slope. In our experiments for convex learners, we fix the fraction of injected poisoning points to 0.1, as by Gu et al. [1], and we report a dot for each combination of λ and γ as specified in Sect. 3.1. Figures 5, 6 and 7 show the relationship between the backdoor slope and the backdoor effectiveness, measured as the percentage of samples with the trigger that mislead the classifier, respectively for MNIST, CIFAR10 and Imagenette. We report the accuracy on the clean test dataset (TS) and the test dataset with the backdoor trigger (TS+BT). For the RBF SVM, we report the accuracy for two different γ values.

Interestingly, our plots show a region where the accuracy of the classifiers on benign samples is high, yet the classifier exhibits low accuracy on samples with triggers. For linear classifiers, this region equals low-regularized classifiers. In

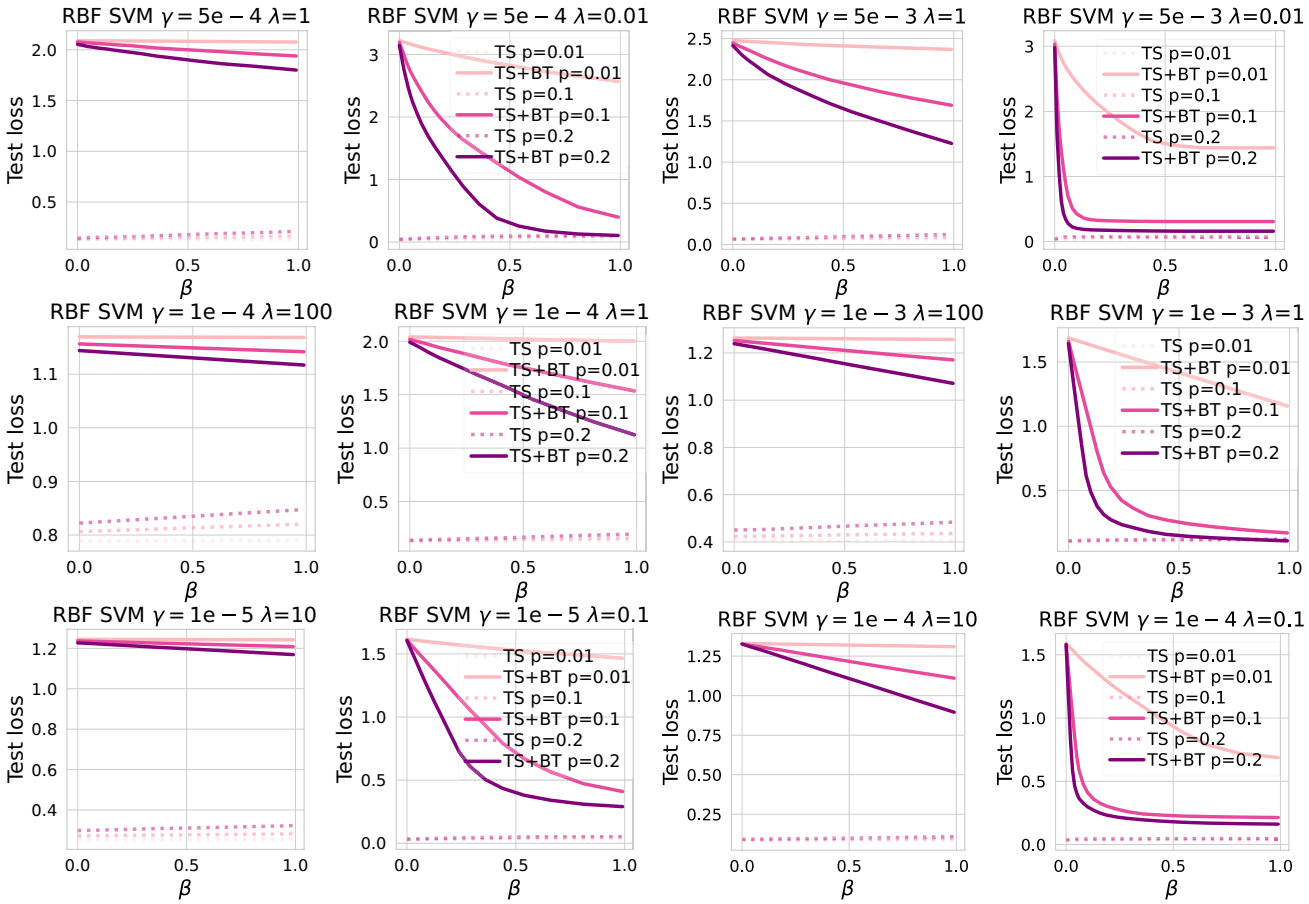


Fig. 4 Backdoor learning curves for MNIST 7 vs 1 (top row), CIFAR10 *airplane vs frog* (middle row) and Imagenette *tench vs truck* (bottom row) when changing the kernel parameter γ on RBF SVM. Darker lines represent a higher fraction of poisoning samples p injected into the training set. We report the loss on the clean test samples (TS) with a dashed line and on the test samples with the backdoor trigger (TS+BT) with a solid line

the case of the RBF SVM, the best trade-off is achieved with high λ (strong regularization) and small γ, which also constrain SVM's complexity. Our results thus indicate that in these cases, the classifier is not flexible enough to learn the backdoor in addition to the clean test samples. Conversely, as long as the classifier has enough flexibility, it can learn the backdoor without sacrificing clean test accuracy. In a nutshell, by choosing the hyperparameters appropriately, we can obtain a classifier able to learn the original task but not the backdoor. However, there is a trade-off between the accuracy of the original task and the robustness of backdoor classification. In Figs. 5, 6 and 7, we extend the comparison between the backdoor learning slope and the attack effectiveness, considering a stronger attack that exploits larger or more visible triggers. With these attacks, the trade-off region is diminished, resulting in fewer viable configurations of hyperparameters that yield a robust model. This result aligns with our earlier findings based on backdoor learning curves: as the attack strength increases, the learning curve descends more steeply, exhibiting a higher backdoor slope. Our results suggest that system designers should thus prioritize maximizing regularization in models, ensuring the tradeoff with accuracy remains acceptable, to deploy a more robust and effective machine learning model against potential backdoor attacks.

As a final check to assess the reliability of the backdoor learning slope, we plot in Fig. 8 clean and backdoor accuracy when training an RBF SVM with different hyperparameter configurations. We train, for each configuration, the classifier on the poisoned dataset. On the top row, we show the results for CIFAR10 *airplane vs frog*, and on the bottom row the results for Imagenette *tench vs truck*. We followed the same backdoor setting for the backdoor slope in Figs. 5, 6 and 7, i.e. trigger size 8×8 for MNIST and trigger visibility $c_m = 75$ for Imagenette. Analogous to our previous findings, there exists a trade-off region where the clean accuracy is high (red), while the backdoor accuracy is low (blue), suggesting a higher

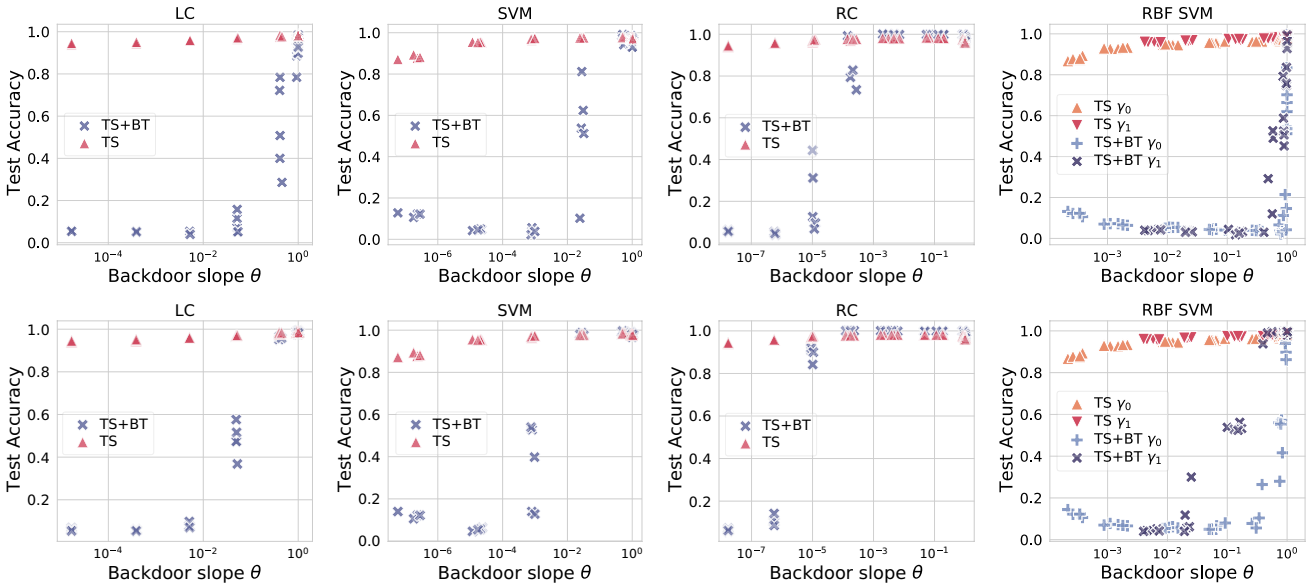


Fig. 5 Backdoor slope θ vs clean accuracy (red) and backdoor effectiveness (blue) on MNIST 7 vs. 1 with backdoor trigger size 3×3 (top row) and 6×6 (bottom row). We measure the classification accuracy on the untainted test samples (TS), and on the same samples after injecting the backdoor trigger (TS+BT). We chose the γ parameter for the RBF kernel as $\gamma_0 = 5\text{e}{-04}$ (orange triangle for clean data, light blue plus for data with trigger) and $\gamma_1 = 5\text{e}{-03}$ (red inverted triangle for clean data, dark blue x for data with trigger)

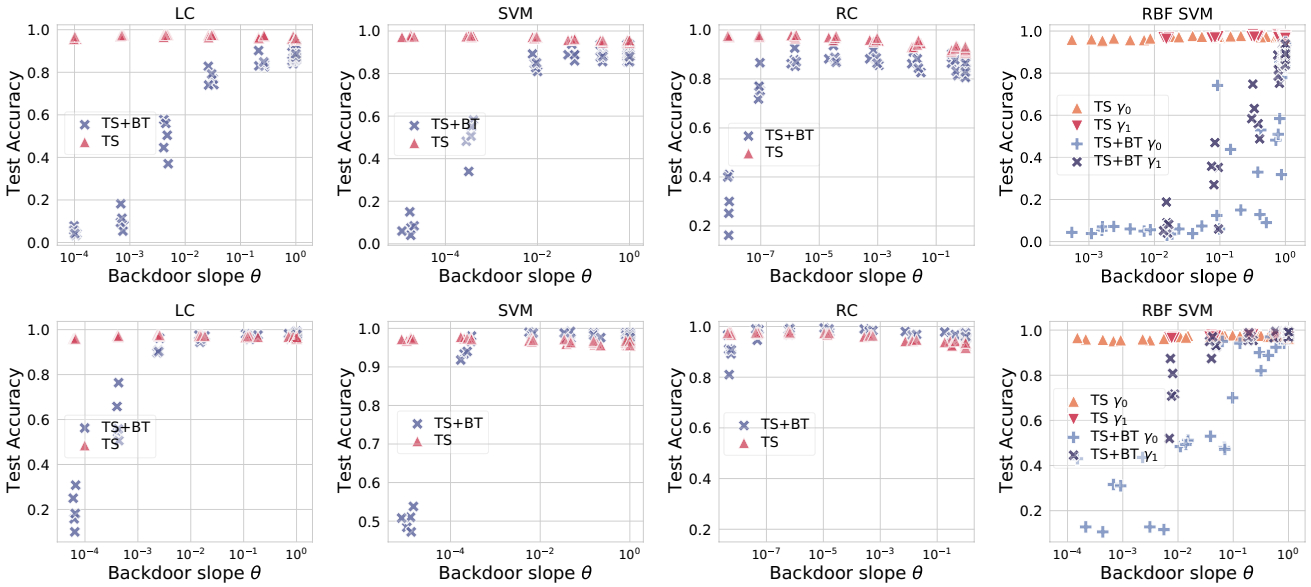


Fig. 6 Backdoor slope θ vs clean accuracy (red) and backdoor effectiveness (blue) on CIFAR10 *airplane vs frog* with backdoor trigger size 8×8 (top row), and 16×16 (bottom row). We measure the classification accuracy on the untainted test samples (TS), and on the same samples after injecting the backdoor trigger (TS+BT). We chose the γ parameter for the RBF kernel as $\gamma_0 = 1\text{e}{-04}$ (orange triangle for clean data, light blue plus for data with trigger) and $\gamma_1 = 1\text{e}{-03}$ (red inverted triangle for clean data, dark blue x for data with trigger)

robustness against backdoors. Analogous to the backdoor slope, the best region is obtained with reduced complexity, thus regularizing it or reducing γ. Consequently, we conclude that smaller values of γ can effectively guide the model toward a more robust configuration against backdoor attacks. Nevertheless, it also emerges that the most decisive factor influencing model robustness remains the regularization term.

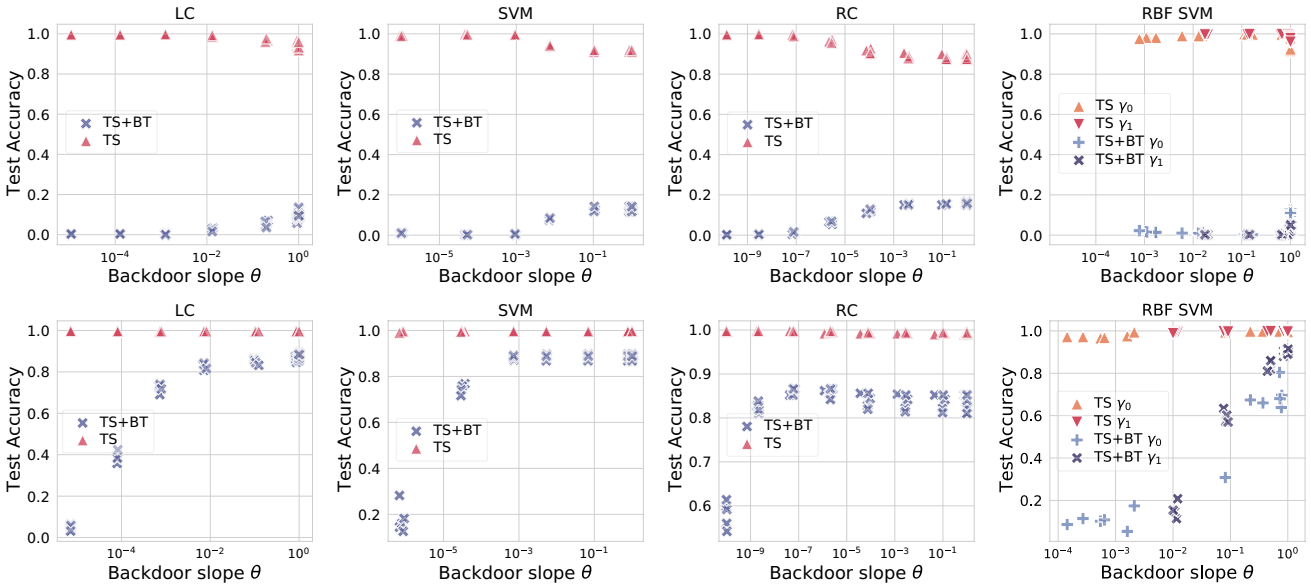


Fig. 7 Backdoor slope θ vs clean accuracy (red) and backdoor effectiveness (blue) on Imagenette *tench vs truck* with trigger visibility $c_m = 10$, i.e. almost imperceptible, (top row) and $c_m = 75$ (bottom row). We measure the classification accuracy on the untainted test samples (TS), and on the same samples after injecting the backdoor trigger (TS+BT). We chose the γ parameter for the RBF kernel as $\gamma_0 = 1e{-}05$ (orange triangle for clean data, light blue plus for data with trigger) and $\gamma_1 = 1e{-}04$ (red inverted triangle for clean data, dark blue x for data with trigger)

While this measure works well on convex learners, its roots in influence functions prevent a direct application on deep neural networks. As pointed out in [34] the analytical gradient in Eq. 5 at $\beta = 0$ is unstable for deep neural networks. To overcome this deficiency, we estimate it with finite difference approximation, obtaining:

$$\left.\frac{\partial L}{\partial \beta}\right|_{\beta=0} = \frac{L(\mathcal{P}_{\text{ts}}, \boldsymbol{w}^\star(h)) - L(\mathcal{P}_{\text{ts}}, \boldsymbol{w}^\star(0))}{h}. \tag{7}$$

We report the the results for Resnet18 and Resnet50 in Table 1 where we used $h = \{0.01, 0.1, 0.2, 1\}$. For each combination of poisoning percentage and number of epochs, we report the estimate of the backdoor learning slope when choosing different h values. The closer h is to 0, the closer to 1 is the backdoor slope of the neural network. This result is consistent with Fig. 3, where the backdoor learning curves drop similarly fast, suggesting a high vulnerability of the model in the presence of backdoor samples. A subtle difference is that when increasing h, there is more evidence for higher vulnerability of neural networks trained with more epochs or when increasing the percentage of poisoning points.

Remarks on model selection/hyperparameter tuning. The observed degradation of accuracy on the backdoored samples (blue dots) at lower slopes in Figs. 5, 6 and 7 suggests that models with a minimal backdoor learning slope, induced by a strong regularization, retain the ability to remain robust. Quite surprisingly, this phenomenon does not have a substantial impact on the classification accuracy of the pristine samples (red triangles), highlighting an advantageous trade-off for defending against backdoor attacks. In other words, there is a wide region in the hyperparameter space in which the model still keeps a very high accuracy on clean data, but it is essentially unable to learn the backdoor samples. This trade-off enables the defender to find a sweet spot in which the model can be sufficiently robust to backdoor attacks. In practice, this means that well-regularized models can be made resilient to backdoor attacks with a negligible impact on classification accuracy, by simply performing an appropriate choice of the hyperparameters, including the regularization term λ, number of epochs or neurons. In particular, such hyperparameters can be tuned to regularize the learning process as much as possible, while retaining an acceptable classification accuracy for the task at hand. We conclude this section by pointing out that this finding is all but trivial, as also highlighted in recent work: Bagdasaryan and Shmatikov [35] claim that using well-regularized models is the only *effective* and *applicable* defense for systems that are deployed and maintained in practical machine-learning operations (MLOps) pipelines, as any other defensive technique will require significant and costly changes to the deployment pipeline infrastructures, inducing a high technical debt on future system maintenance.

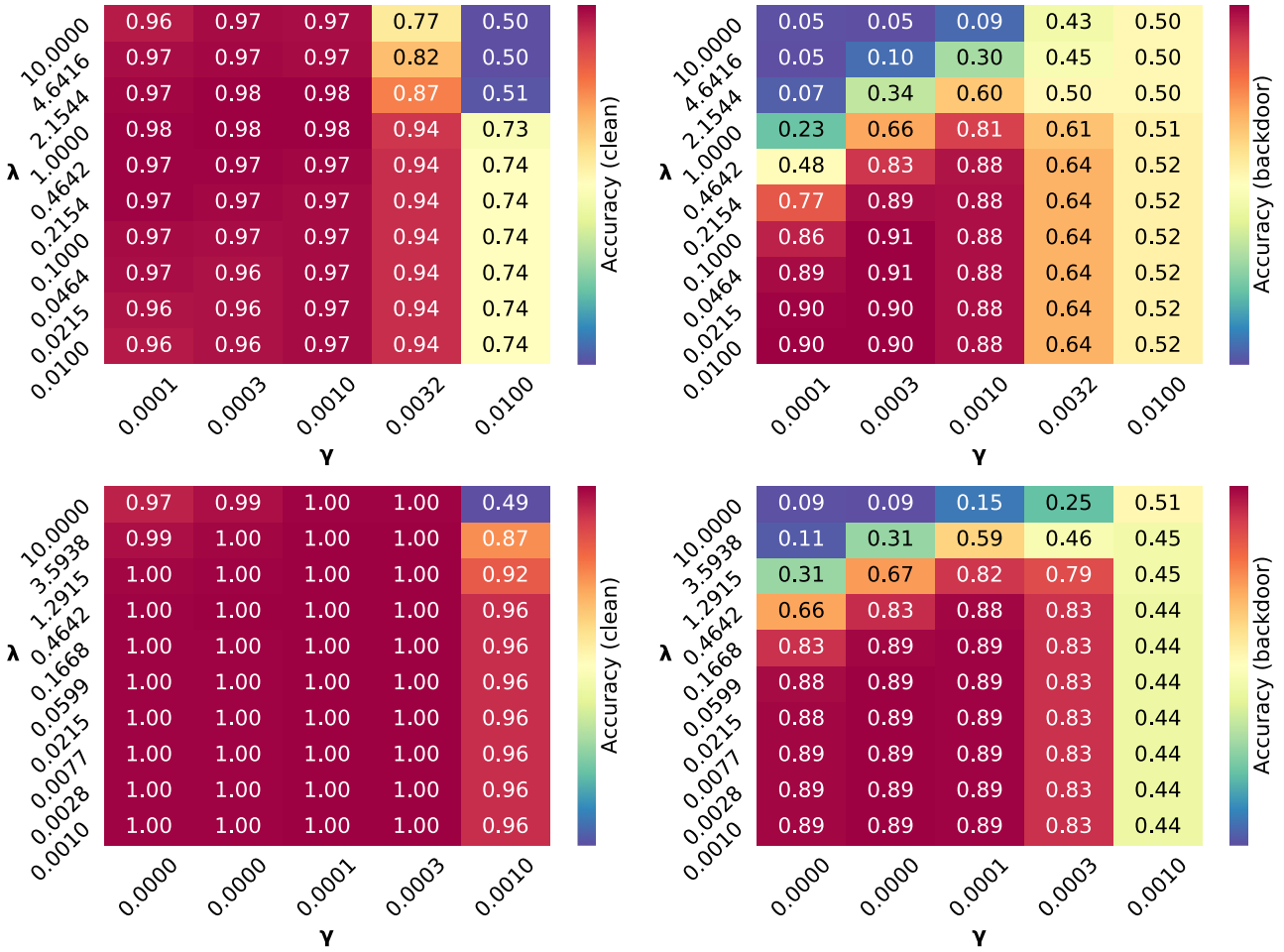


Fig. 8 Influence of γ (x-axis) and λ (y-axis) on the backdoor effectiveness (right) and clean accuracy (left) for CIFAR10 *airplane vs frog* (top row) and Imagenette *tench vs truck* (bottom row). The Backdoor is mounted with trigger size 8×8 for CIFAR10 and visibility $c_m = 75$ for Imagenette. The plots show that there are hyperparameter configurations for which clean accuracy is high (red regions on the left plots), while the accuracy on the backdoored points is low (blue regions on the right plots)

Table 1 Backdoor learning slope for Resnet18 and Resnet50 when increasing the percentage of backdoor poisoning p, the number of epochs *(#Epochs)*, and parameter h for estimate in Eq. 7

Model	p	#Epochs	Slope h=0.01	Slope h=0.1	Slope h=0.2	Slope h=1	Accuracy TS+BT	Accuracy TS
Resnet18	0.05	10	0.9955	0.9872	0.9752	0.9026	0.4163	0.9588
Resnet50	0.05	10	0.9965	0.9895	0.9785	0.9169	0.7197	0.9781
Resnet18	0.05	50	0.9986	0.9900	0.9797	0.9281	0.5256	0.9737
Resnet50	0.05	50	0.9992	0.9936	0.9849	0.9377	0.8067	0.9881
Resnet18	0.15	10	0.9955	0.9878	0.9774	0.9189	0.8804	0.9568
Resnet50	0.15	10	0.9966	0.9902	0.9943	0.9231	0.9440	0.9826
Resnet18	0.15	50	0.9987	0.9937	0.9864	0.9384	0.8893	0.9720
Resnet50	0.15	50	0.9992	0.9939	0.9971	0.9403	0.9509	0.9890

We also report the corresponding backdoor effectiveness *(Accuracy TS+BT)* and clean accuracy *(Accuracy TS)*, measures respectively as the percentage correctly classified test samples with and without the backdoor trigger

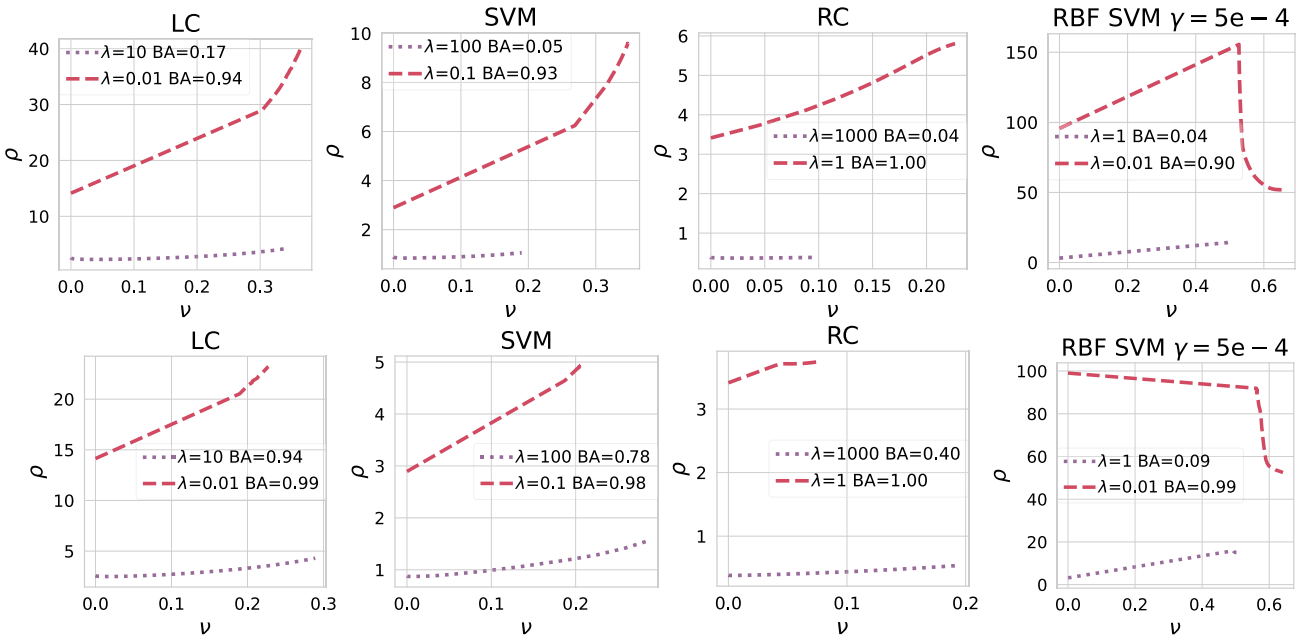


Fig. 9 Backdoor weights deviation for the logistic classifier (LC), support vector machine (SVM), the ridge classifier (RC) and SVM with RBF kernel on MNIST 7 vs. 1 poisoned with backdoor trigger [1]. We report the results for trigger size 3×3 (top row) and 6×6 (bottom row). We specify the regularization parameter λ and backdoor accuracy (BA) for each setting in the legend of each plot

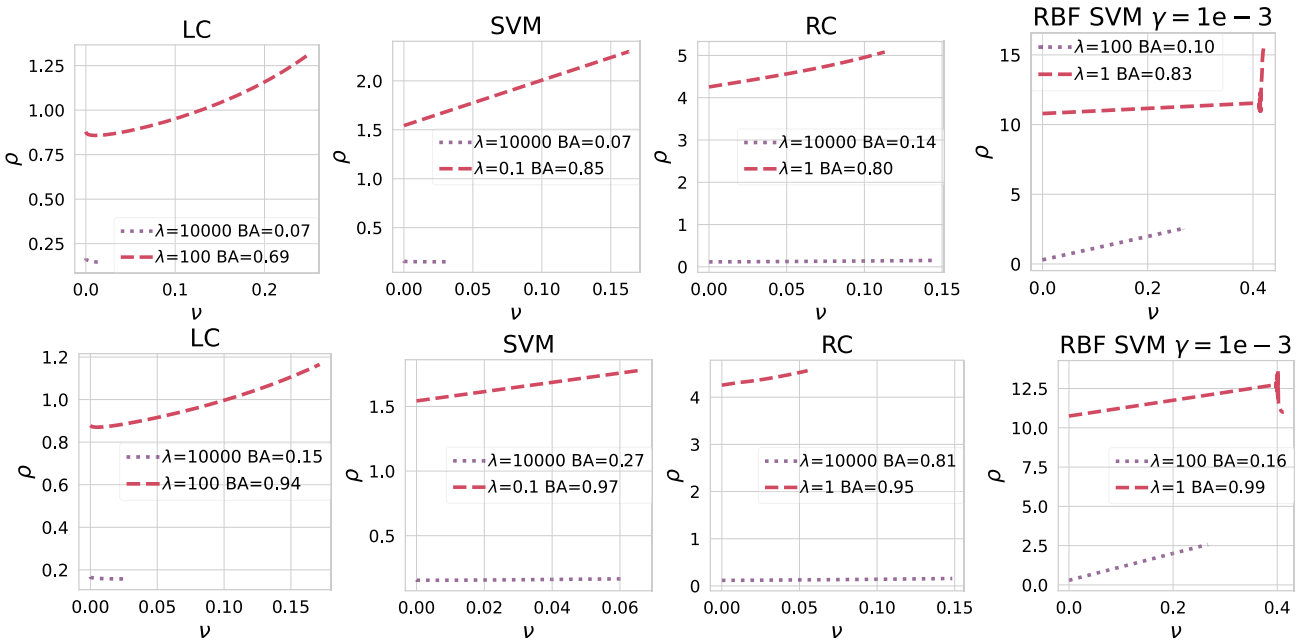


Fig. 10 Backdoor weights deviation for the logistic classifier (LC), support vector machine (SVM), the ridge classifier (RC) and SVM with RBF kernel on CIFAR10 *airplane vs frog* poisoned with backdoor trigger [1]. We report the results for trigger size 8×8 (top row) and 16×16 (bottom row). We specify the regularization parameter λ and backdoor accuracy (BA) for each setting in the legend of each plot

3.2.3 Empirical Parameter Deviation Plots

After having investigated which factors influence backdoor effectiveness, we shift our focus to examining how the model's weights change during the training process when the dataset is tainted with backdoor samples. We aim to determine whether there is an increase in complexity or not.

We use our two measures proposed in Sect. 2, ρ and ν to analyze the parameter change. The former, ρ, monitors the change of the weights, for example, whether they increase or decrease. The latter, ν, measures the change in orientation or angle of the classifier. We plot both measures with different regularization parameters, trigger size, or visibility with a fraction of poisoning points to $p = 0.1$ in Figs. 9, 10 and 11. Within each plot, we also report the backdoor accuracy (BA) representing the model's performance on backdoor samples at the end of training.

On linear classifiers, $\rho(\boldsymbol{w})$ increases during the backdoor learning process. This equals an increase in the weights' values, suggesting that the classifiers become more complex while learning the backdoor. However, when investigating the RBF SVM, the results are slightly different. Indeed, when increasing γ and decreasing λ, the classifier becomes flexible and complex enough to learn the backdoor without increasing its complexity. On the other hand, when decreasing γ, the model is constrained to behave similarly to a linear classifier. In this way, analogously to linear classifiers, the model needs to increase its complexity to learn the backdoor. When increasing the trigger size or visibility the results are similar, thus confirming the previous analysis. However, as a result of increasing the attacker's strength, the backdoor accuracy turns out to be higher.

3.2.4 Explaining backdoor predictions

In the following, we give a graphical interpretation of the poisoned convex-classifier's decision function, expressed by its internal weights, for which interpretation of their results is easier [26, 27]. We consider the results for a backdoor trigger [1] in a specific position, as its influence on the classifier decision is easier to see. Conversely, the backdoor trigger by for example Zhong et al. [32] spans the entire image, and therefore its influence is harder to spot from the interpretability plots. In particular, given a sample x we aim to compute and show the gradient of the classifier's decision function with respect to x. We use an SVM with regularization $\lambda = 1\text{e}{-02}$ for MNIST 7 vs 1 and CIFAR10 *airplane vs frog*, and report the results in Fig. 12. For MNIST, we consider the digit 7 with the trigger, showcasing the gradient of the clean classifier's decision function. We present the results of the gradient from the clean and poisoned classifiers corresponding to the clean and backdoored inputs. Since we train a linear classifier on the input space, the derivative coincides with the classifier's weights. Intriguingly, the classifier's

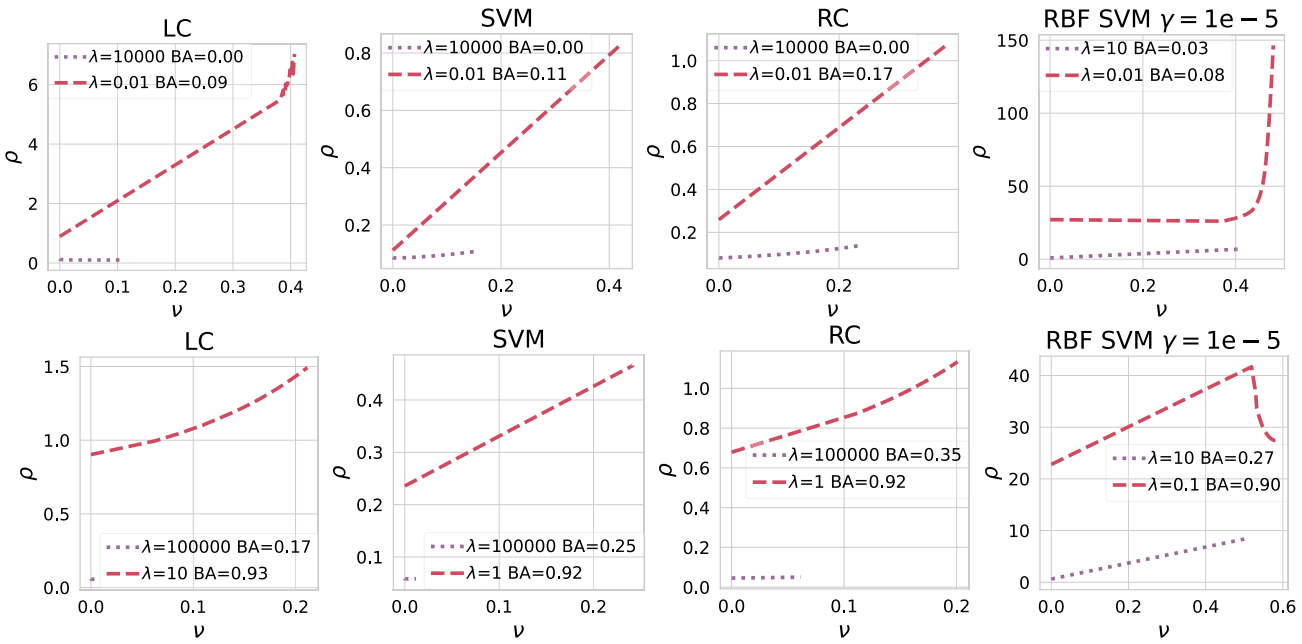


Fig. 11 Backdoor weights deviation for the logistic classifier (LC), support vector machine (SVM), the ridge classifier (RC) and SVM with RBF kernel on Imagenette *tench vs truck* poisoned with backdoor trigger [32]. We report the results for visibility $c_m = 10$ (top row) and $c_m = 75$ (bottom row). We specify the regularization parameter λ and backdoor accuracy (BA) for each setting in the legend of each plot

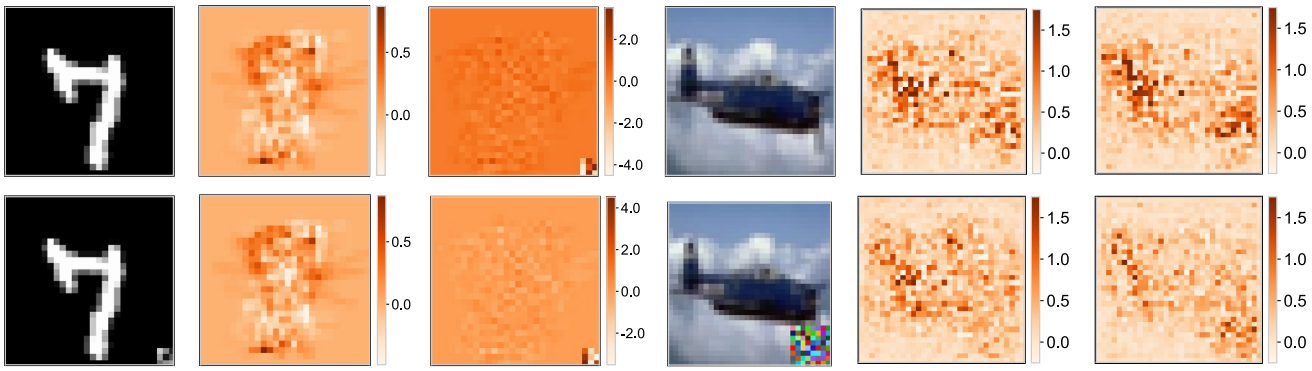


Fig. 12 Input gradients of untainted and poisoned SVMs on pristine (top row) and backdoored (bottom row) test samples. Each row shows two sets of three images. Each set contains an example from MNIST *7 vs 1* or CIFAR10 *airplane vs frog* (left), along with the corresponding input gradient of the untainted SVM (middle), and of the poisoned SVM (right). For CIFAR10, we consider the maximum gradient of each pixel among the three channels

Fig. 13 Influential training points for a high-complexity classifier. Considering an SVM with $\lambda = 0.01$ trained on MNIST, and with $\lambda = 0.1$ trained on CIFAR10, and Imagenette, we show the top 7 most influential training samples on the prediction of the samples with the red border

weights increase in magnitude and now exhibit high values in the bottom right corner, where the trigger is located. From CIFAR10, we show a poisoned airplane. We report the gradient mask obtained by considering the maximum value for each channel, both for the clean and backdoored classifier. Also, in this case, the backdoored model shows higher values in the bottom right region, corresponding to the trigger location. This means that the analyzed classifiers assign high importance to the trigger to discriminate the class of the input points.

Summarizing, the plots in Fig. 12 further confirm our findings regarding the change of the internal parameters during the backdoor learning process. In particular, we have seen that less regularized classifiers need to increase their weights and thus complexity to learn the backdoor. Conversely, when the flexibility of the classifier increases then it can learn the backdoor easier without significantly altering its complexity.

3.2.5 Visualizing influential training data points

Influence functions are used in the context of ML to identify the training points more responsible for a given prediction [13]. In Sect. 2 we have seen how they represent the basis of our backdoor learning slope measure. In this section, we employ them to show their outcomes and provide further insight into the relationship between complexity and backdoor effectiveness. To this end, as in Sect. 3.1, we poison 10% of the training dataset. According to previous experiments, we employed the backdoor trigger in [1] for MNIST and CIFAR10 with trigger size 3×3 and 6×6 respectively, while for Imagenette we employed the trigger in Zhong et al.

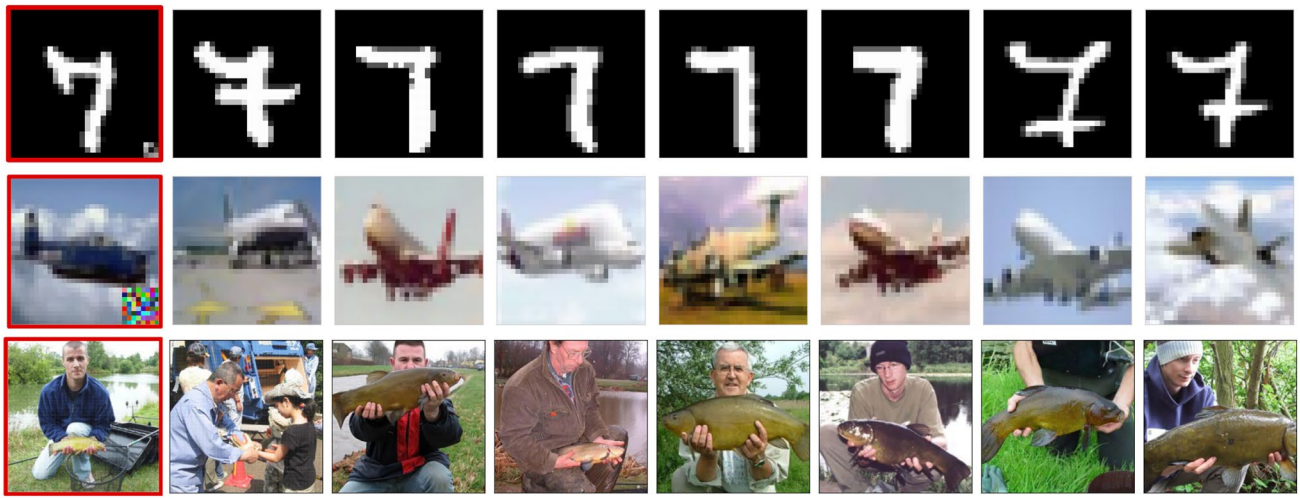


Fig. 14 Influential training points for low-complexity classifiers. Considering an SVM with $\lambda = 1e-3$ trained on MNIST, and with $\lambda = 1e-5$ trained on CIFAR10, and Imagenette, we show the top 7 most influential training samples on the prediction of the samples with the red border

[32] with higher visibility (i.e. $c_m = 75$). In Figs. 13 and 14, considering respectively a high- and a low-complexity classifier, we report the seven most influential training samples on the classification of a randomly chosen test point. For high-complexity classifiers, many of these training samples contain the trigger. In contrast, this is not the case for low-complexity classifiers. These results suggest that low-complexity classifiers rely less on the samples containing the backdoor trigger in their predictions.

4 Related work

We first review the literature about backdoor poisoning attacks and defenses. Afterward, we focus on defenses that increase the robustness against backdoors by reducing the model's complexity. We conclude the section by discussing the relationship between our proposed framework and influence functions.

Backdoor poisoning. Although backdoors were introduced recently [1, 3, 6], a plethora of backdoor attacks and defenses have been published. For a more detailed overview, we refer the reader to surveys in this area [3, 4, 36]. Despite the quickly-growing literature about this topic, the majority of the previous works [21, 33, 37, 38] study different types of poisoning attacks, i.e., not backdoors. In contrast, only a few works have studied factors that influence the success of this attack. Baluta et al. [39] and [40] studied the relationship between backdoor effectiveness and the percentage of backdoored samples. Salem et al. [41] experimentally investigated the relationship between the backdoor effectiveness and the trigger size. Similarly, Severi et al. [42] have analyzed the correlation between the backdoor success and the attacker's strength on malware classifiers. Schwarzschild et el. [43] evaluated the performance of backdoor attacks when scaling the dataset size while fixing the poison budget. Finally, Li et al. [44] demonstrated that the backdoor performance is sensitive to the location of the trigger on the attacked image. We instead do not limit our study to neural networks but also study other models. Furthermore, we also investigate other relevant factors, e.g., regularization and visibility, and their interaction at once.

Complexity and backdoor defenses. In this work, we have analyzed the relationship between backdoor effectiveness and different factors, including complexity, controlled via regularization and the RBF kernel's hyperparameter. In this study, we have demonstrated that reducing complexity by choosing appropriate hyperparameter values improves robustness against backdoors. Our findings align with the insights presented in Frnay et al. [45], who suggested that overfitting avoidance techniques like, e.g., regularization, can offer partial mitigation against random label noise [46, 47]. Expanding upon their discourse, we apply and extend this consideration to the context of backdoor attacks, wherein the noise is intentionally and strategically introduced to deceive the machine learning model. Some of the defenses proposed against backdoors use different techniques to reduce complexity. These techniques include pruning [48, 49], data augmentation [50, 51] and gradient shaping [52]. However, from these works, it remains unclear why reducing complexity alleviates the threat of backdoor poisoning. To the best of our knowledge, our work is the first to investigate this aspect.

Relation to influence functions. Influence functions originated in robust statistics [53] and were later used as a tool to measure the influence of specific training points on the

classification output [13, 54]. In our work, we clarify that influence functions naturally descend from the incremental learning formulation in Eq. 1, showing that they quantify the velocity with which the classifier will learn new points. As seen in Sect. 2, they correspond to the partial derivative of the learning curve at the point $\beta = 0$. Moreover, we leveraged them by proposing a measure, namely the backdoor slope, which quantifies the ability of a classifier to learn backdoors. This measure allowed us to study the factors that impact backdoor effectiveness.

Several defense approaches confirm that the influence functions, or gradients during training, are indeed related to backdoor learning. For example, some defenses are directly based on the gradient [55], based on gradient differences [56, 57], or based on differential privacy that noises the gradients during training [52, 58, 59].

5 Conclusions, limitations and future work

In this paper, we presented a framework to analyze the factors influencing the effectiveness of backdoor poisoning. We carried out experiments on convex learners, also used in transfer-learning scenarios, and neural networks. As in previous work [7, 13], we focus our analysis on two-class classification problems for convex learners, and on multiclass classification when considering neural networks.

Our analysis shows that the effectiveness of backdoor attacks inherently depends on (i) the complexity of the target model, (ii) the fraction of backdoor samples in the training set, and (iii) the size and visibility of the backdoor trigger. By analyzing the influence of the first factor on backdoor learning, we are the first to unveil a region in the hyperparameter space where the accuracy on clean test samples remains high while the accuracy on backdoor samples is low. Specifically, we discovered that the target model needs to significantly increase the complexity of its decision function to learn backdoors, which is only possible when the model is not regularized enough. Conversely, when raising the model's regularization, we can keep high performance on clean samples and be unaffected by potential backdoor attacks. However, increasing the attacker's strength, i.e., the last two factors, makes the attack more effective, shrinking this region and thus exposing the model to greater vulnerability. We, therefore, conclude that a prudent strategy to preserve robustness against potential poisoning attacks is to regularize as much as possible during the hyperparameter optimization phase, thereby reducing the backdoor learning slope while ensuring that the trade-off with accuracy remains acceptable.

The study of more factors, like, for example, the dimensionality of the data, is straightforward using the proposed framework but left for future work. Our current results already provide important insights and provide a starting point to derive guidelines for designing models that are more robust against backdoor poisoning.

Appendix A: Datasets

The MNIST dataset [18] contains 70,000 observations representing 28×28 grayscale images of handwritten digits from 0 to 9. The CIFAR10 dataset [19] contains 60,000 colour images of size 32×32 pixels divided in 10 classes, each with 6000 observations. Finally, the Imagenette dataset [20] is a subset of 10 classes (i.e., tench, English springer, cassette player, chain saw, church, French horn, garbage truck, gas pump, golf ball, parachute) from Imagenet. We use the 320px version, where the shortest side of each image is resized to that size.

Appendix B: Additional Experimental Results

In the paper, we have shown the backdoor learning curves only for some classifiers. Here, we report them for all the classifiers considered in this work. As we will discuss later in this section, these results confirm the ones obtained in the paper. In particular, here we consider:

- Support vector machine (SVM) with $\lambda \in \{100, 0.1\}$ for MNIST, $\lambda \in \{10000, 0.1\}$ for CIFAR10, and $\lambda \in \{100000, 1\}$ for Imagenette.
- Ridge classifier (RC) with $\lambda \in \{1000, 1\}$ for MNIST, $\lambda \in \{10000, 1\}$ for CIFAR10, and $\lambda \in \{100000, 1\}$ for Imagenette.
- Logistic classifier (LC) with $\lambda \in \{10, 0.01\}$ for MNIST, $\lambda \in \{10000, 100\}$ for CIFAR10, and $\lambda \in \{100000, 10\}$ for Imagenette.
- SVM with an RBF kernel, where $\lambda \in \{1, 0.01\}$ and $\gamma = 5\text{e}{-04}$ for MNIST, $\lambda \in \{100, 1\}$ and $\gamma = 1\text{e}{-03}$ for CIFAR10, and $\lambda \in \{10, 0.1\}$ and $\gamma = 1\text{e}{-05}$ for Imagenette.

Moreover, we compare the results obtained on the class pairs considered in the paper (7 vs 1 on MNIST, *airplane vs frog* on CIFAR10 and Imagenette *tench vs truck*) with the ones obtained on different pairs.

Backdoor learning curves and backdoor learning slope. In Figs. 15, 16, 17, 18, 19 and 20 we report the backdoor learning curves for each classifier and dataset pair. In Figs. 21, 22 and 23, we report the backdoor learning slope, computed with $p = 0.1$, for all the considered classifiers and all subset pairs. The results do not show significant variation with respect to the ones reported in the paper.

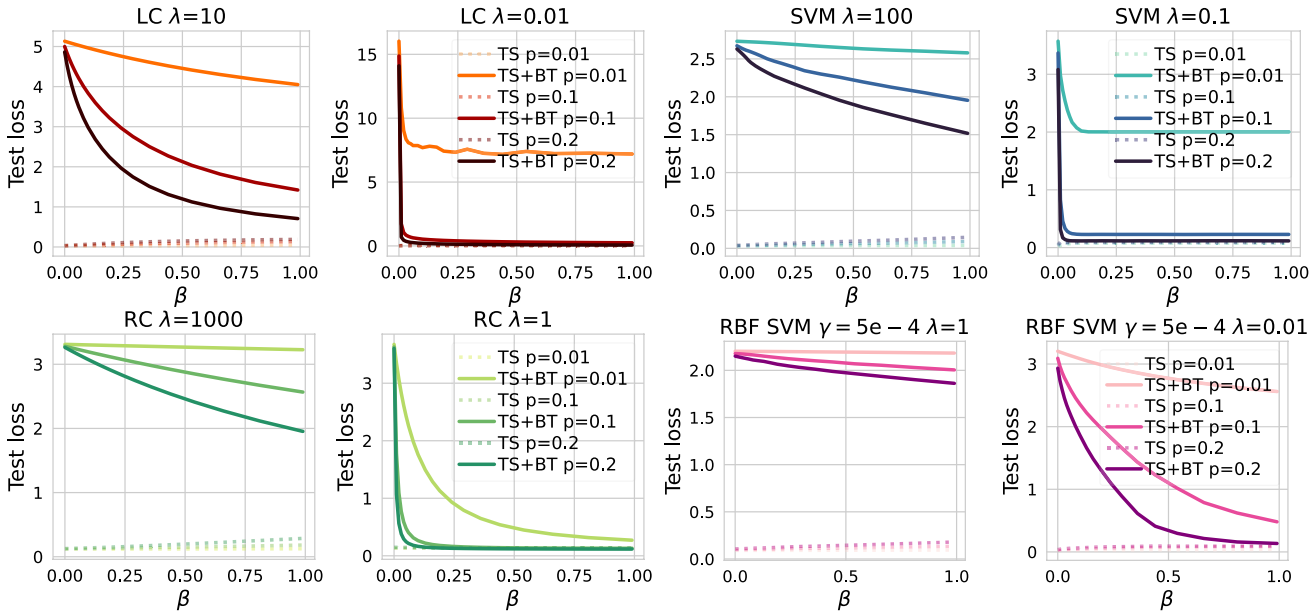


Fig. 15 Backdoor learning curves for different classifiers trained on MNIST 3-0. Darker lines represent a higher fraction of poisoning samples p injected into the training set. We report the loss on the clean test samples (TS) with a dashed line and on the test samples with the backdoor trigger (TS+BT) with a solid line

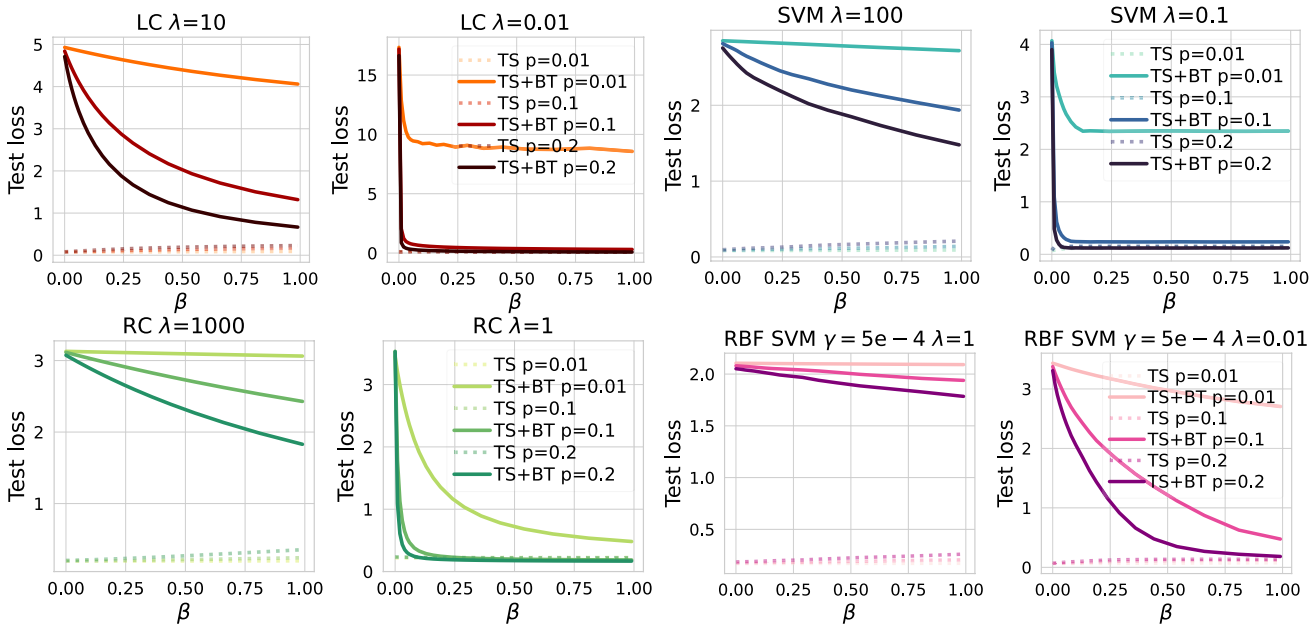


Fig. 16 Backdoor learning curves for different classifiers trained on MNIST 5-2. See the caption of Fig. 15 for further details

Empirical parameter deviation plots. In Figs. 24, 25 and 26, shows how the classifiers' parameters change when the classifiers learn the backdoors. This analysis is carried out with $p = 0.1$. The results do not vary significantly across different classifiers and class pairs. The only exception is MNIST 5 vs 2. The untainted classifier is already quite complex; therefore, it does not increase its complexity when it learns the backdoor.

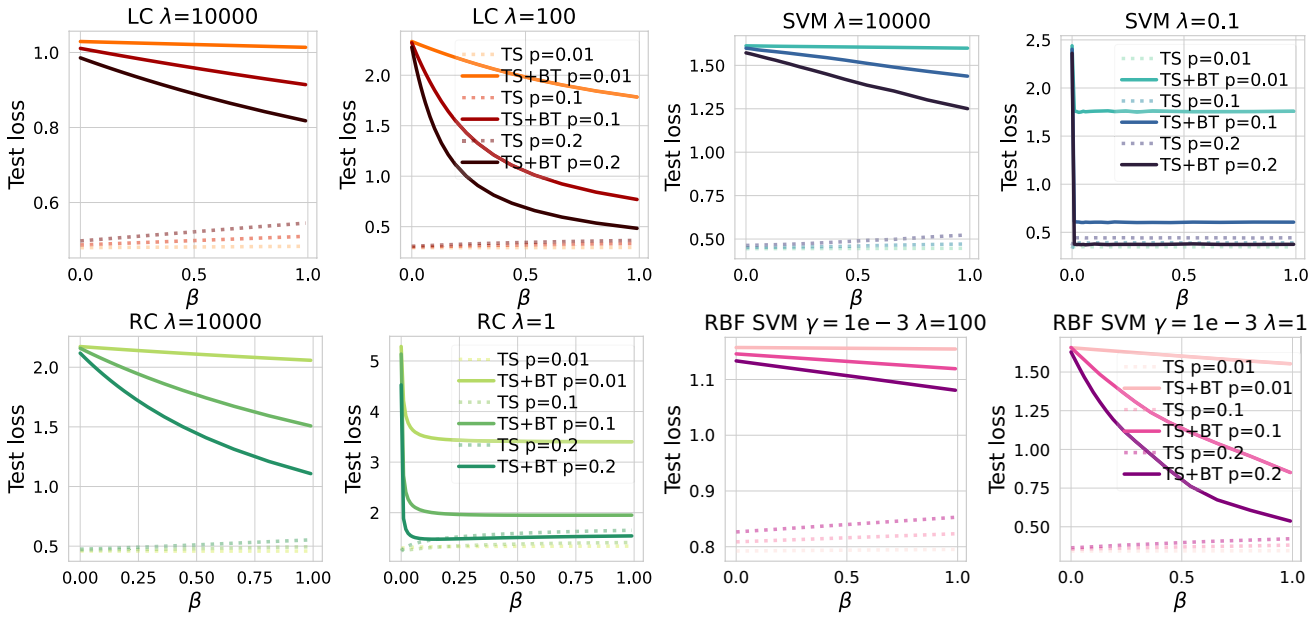


Fig. 17 Backdoor learning curves for different classifiers trained on CIFAR10 *bird vs dog*. See the caption of Fig. 15 for further details

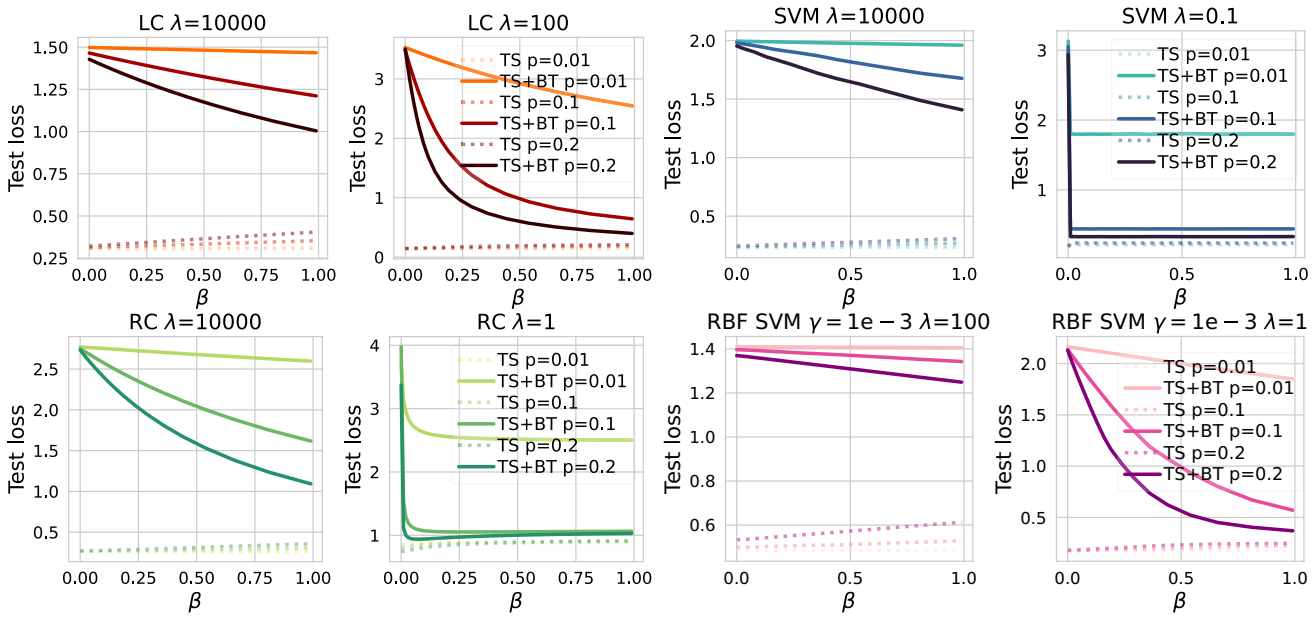


Fig. 18 Backdoor learning curves for different classifiers trained on CIFAR10 *airplane vs truck*. See the caption of Fig. 15 for further details

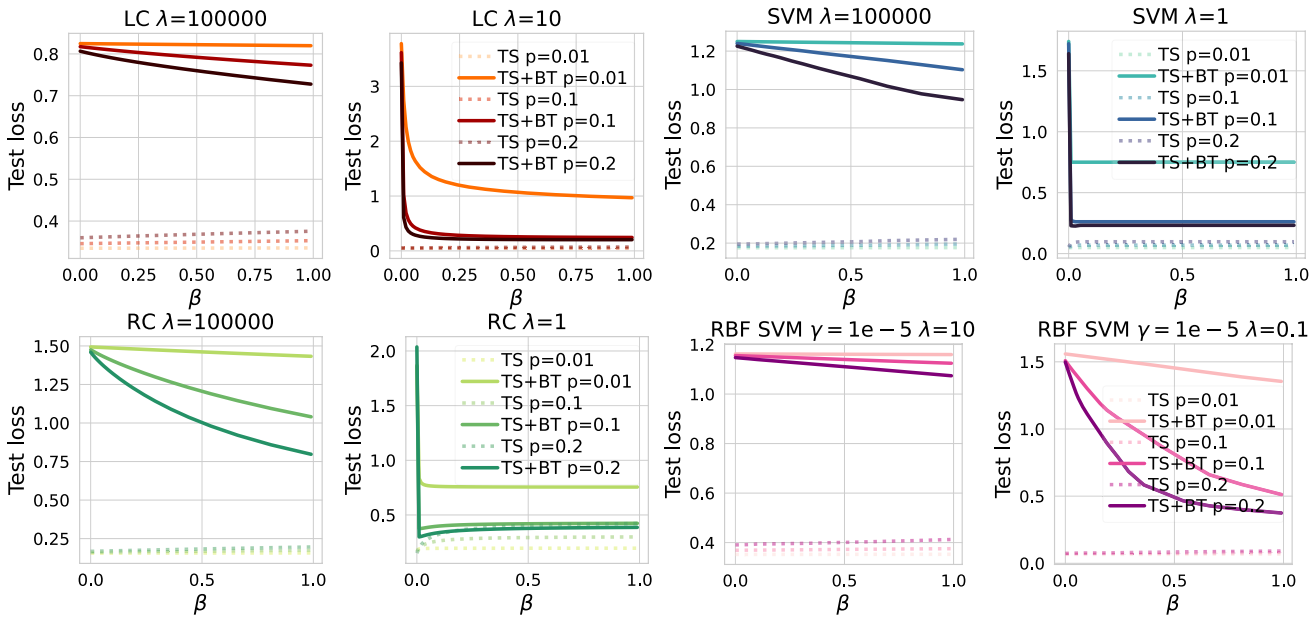


Fig. 19 Backdoor learning curves for different classifiers trained on Imagenette *cassette player vs church*. See the caption of Fig. 15 for further details

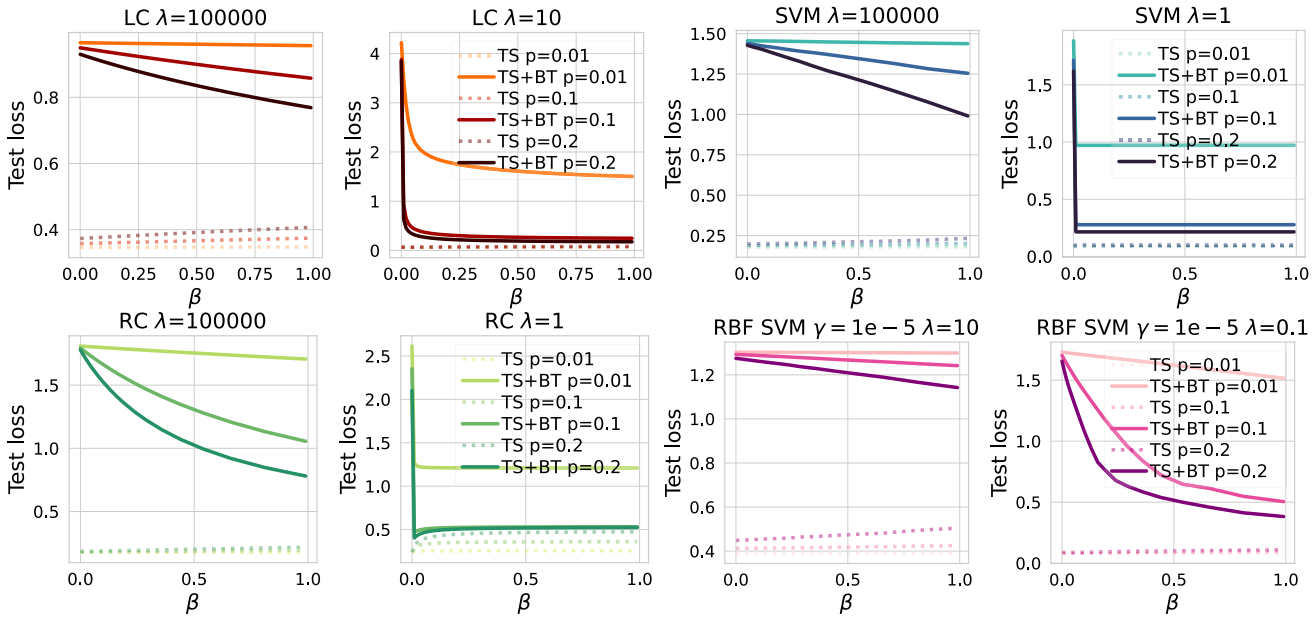


Fig. 20 Backdoor learning curves for different classifiers trained on Imagenette *tench vs parachute*. See the caption of Fig. 15 for further details

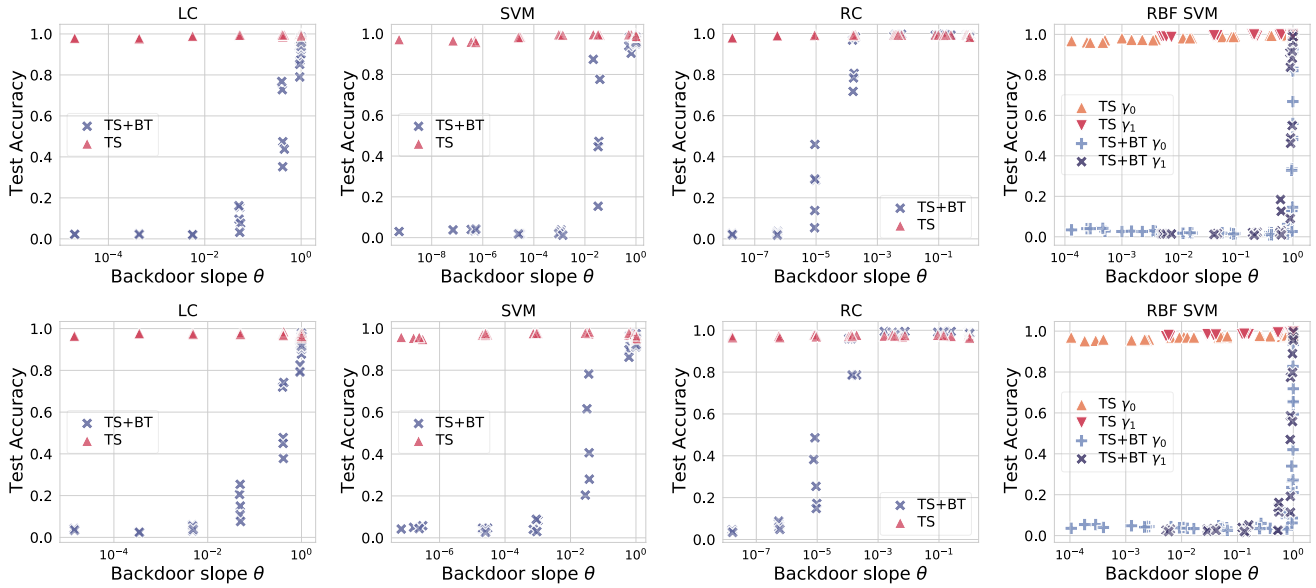


Fig. 21 Backdoor slope θ vs clean accuracy (red) and backdoor effectiveness (blue) on MNIST 3vs.0 (top row) and 5vs.2 (bottom row). We measure the classification accuracy on the untainted test samples (TS), and on the same samples after adding the 3×3 backdoor trigger (TS+BT). We chose the γ parameter for the RBF kernel as $\gamma_0 = 5\text{e}{-04}$ (orange triangle for clean data, light blue plus for data with trigger) and $\gamma_1 = 5\text{e}{-03}$ (red inverted triangle for clean data, dark blue x for data with trigger)

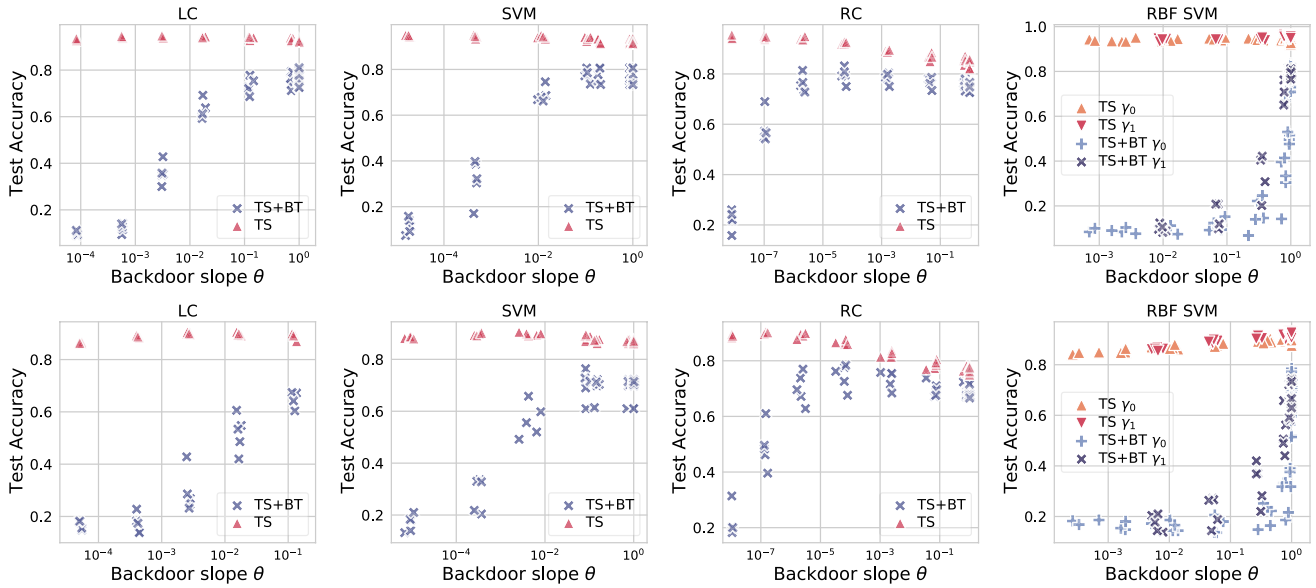


Fig. 22 Backdoor slope vs backdoor (BK) effectiveness on CIFAR10 *airplane vs truck* (top row) and *bird vs dog* (bottom row). See the caption of Fig. 21 for further details. The results are obtained considering a trigger size equal to 8

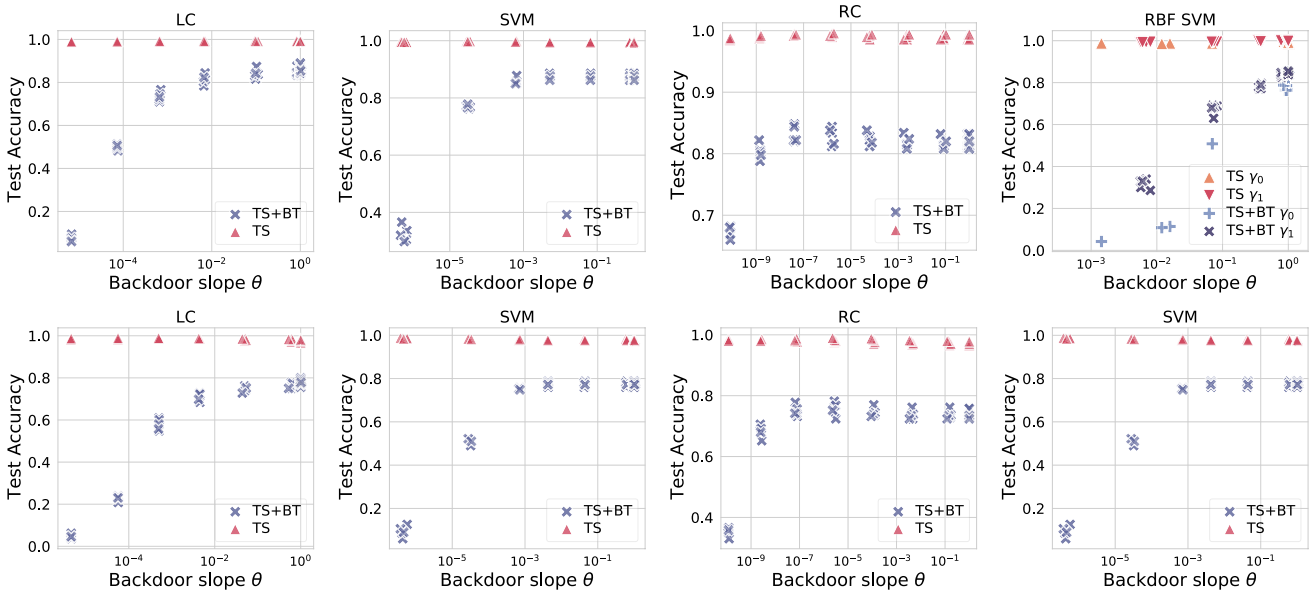


Fig. 23 Backdoor slope vs backdoor (BK) effectiveness on Imagenette *cassette player vs church* (top row) and *tench vs parachute* (bottom row). See the caption of Fig. 21 for further details. The results are obtained considering a trigger size equal to 8

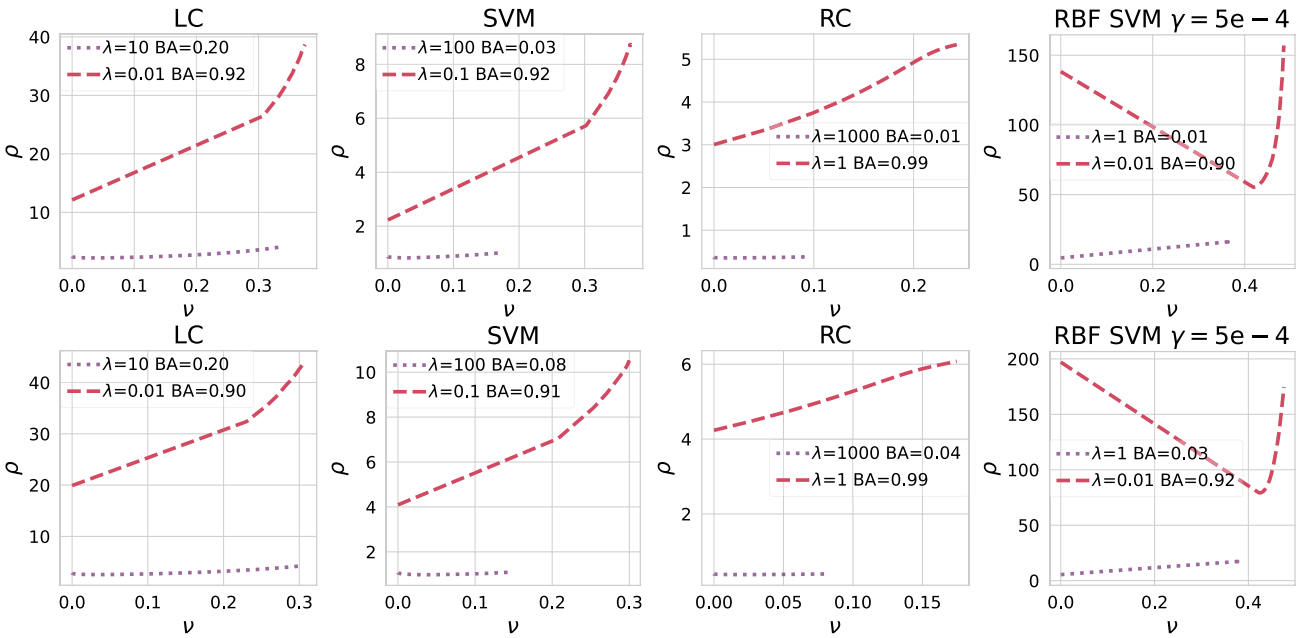


Fig. 24 Backdoor weights deviation for different classifiers trained on MNIST 3 vs 0 (top row) and 5 vs 2 (bottom row). We specify regularization parameter λ and backdoor (BK) accuracy for each setting in the legend of each plot

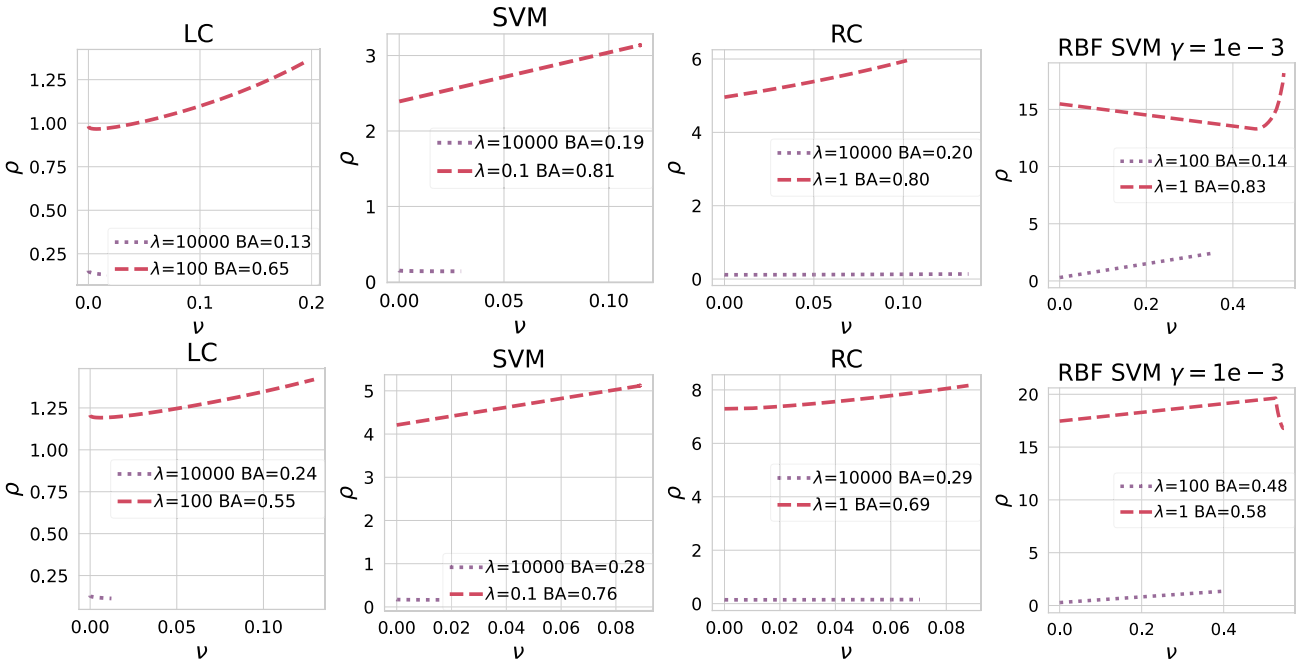


Fig. 25 Backdoor weights deviation for different classifiers trained on CIFAR10 *airplane vs truck* (top), and *bird vs dog* (bottom). See Fig. 24 for further details

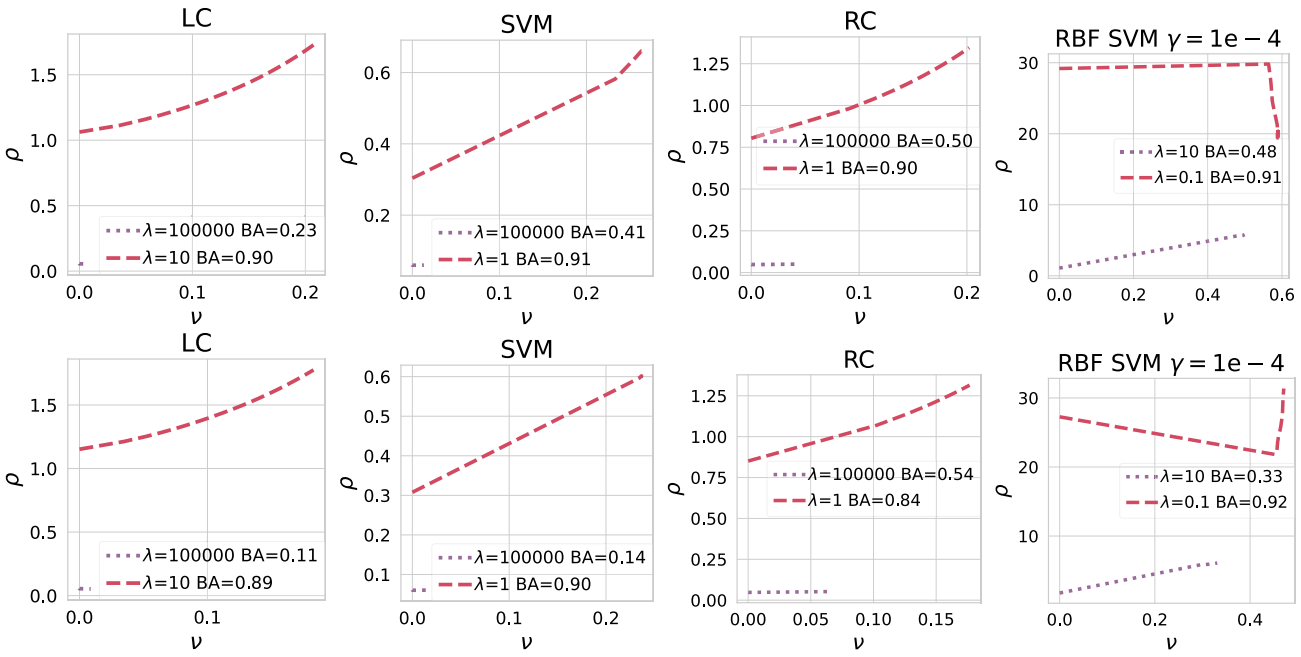
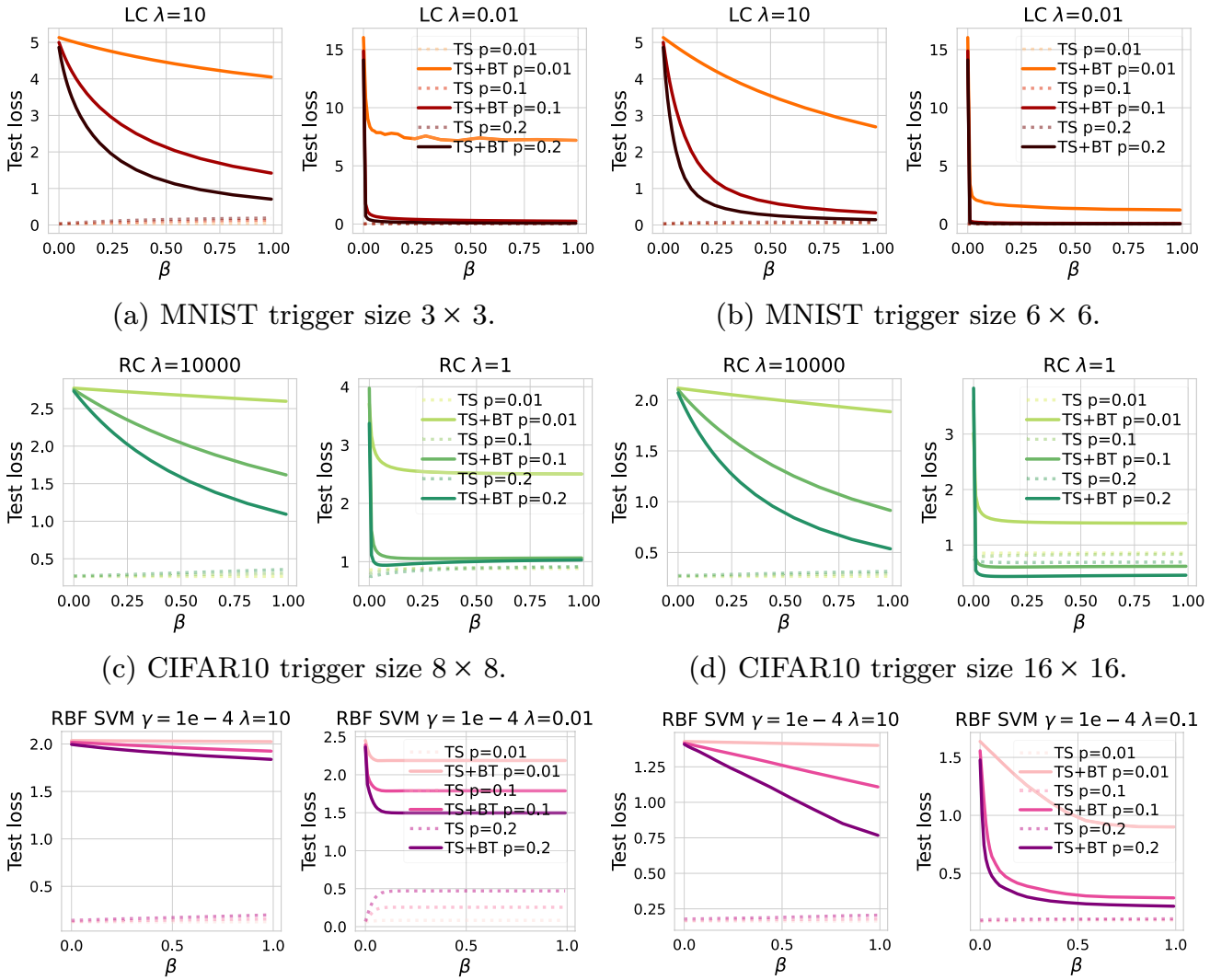


Fig. 26 Backdoor weights deviation for different classifiers trained on Imagenette *tench vs parachute* (top), and *cassette player vs church* (bottom). See Fig. 24 for further details



(a) MNIST trigger size 3×3.

(b) MNIST trigger size 6×6.

(c) CIFAR10 trigger size 8×8.

(d) CIFAR10 trigger size 16×16.

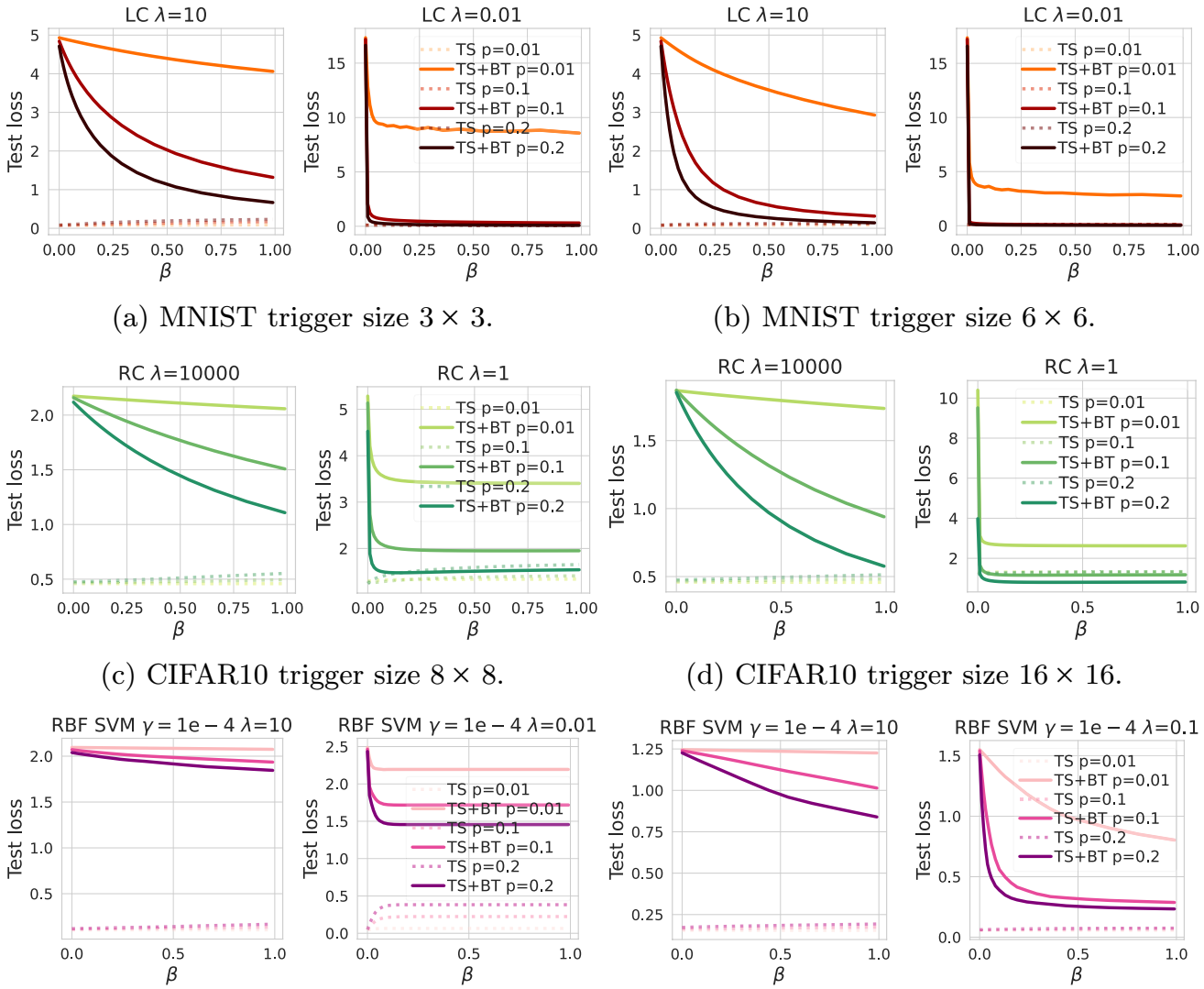
(e) Imagenette trigger visibility $c_m = 10$.

(f) Imagenette trigger visibility $c_m = 75$.

Fig. 27 Backdoor learning curves for: (top row) LC on MNIST 3vs.0 with trigger size 3×3 (left) or 6×6 (right); (middle row) RC on CIFAR10 *airplane vs truck* with trigger size 8×8 (left) or 16×16 (right); (bottom row) RBF SVM on Imagenette *cassette player vs church* with trigger visibility c_m=10 (left) or c_m=75 (right). Further details in Fig. 15

B.1 Increasing the trigger size or visibility

Although it is a known result in the literature that the size of the trigger increases the effectiveness of the attack [7, 41], here, for the first time to the best of our knowledge, we show how it interacts with other factors. In this section we report further experimental results when increasing the trigger size or visibility. As expected, the results in Figs. 27 and 28 show that choosing a larger trigger enhances the effectiveness of the attack. Indeed, when the trigger is larger or more visible the backdoor learning curves go down faster. Using the proposed backdoor slope to analyze the effect of complexity, controlled via the hyperparameters, on the vulnerability against backdoors, we found a region of the hyperparameter space that leads to having desirable performances: an accuracy high on



(a) MNIST trigger size 3×3.

(b) MNIST trigger size 6×6.

(c) CIFAR10 trigger size 8×8.

(d) CIFAR10 trigger size 16×16.

(e) Imagenette trigger visibility $c_m = 10$.

(f) Imagenette trigger visibility $c_m = 75$.

Fig. 28 Backdoor learning curves for: (top row) LC on MNIST 5 vs. 2; (middle row) RC on CIFAR10 *bird vs dog*; (bottom row) RBF SVM on Imagenette *tench vs parachute*

the clean test samples and low on the ones containing the backdoor trigger.

Funding Open access funding provided by Università degli Studi di Genova within the CRUI-CARE Agreement. This work has been partially supported by Spoke 10 "Logistics and Freight" within the Italian PNRR National Centre for Sustainable Mobility (MOST), CUP I53C22000720001; the projects SERICS (PE00000014) and FAIR (PE00000013) under the NRRP MUR program funded by the EU-NGEU; the PRIN 2017 project RexLearn (grant no. 2017TWNMH2), funded by the Italian MUR; and by BMK, BMDW, and the Province of Upper Austria in the frame of the COMET Programme managed by FFG in the COMET Module S3AI.

Data availability The data MNIST, CIFAR10, and Imagenette used in this article can be obtained from the following links: MNIST http://yann.lecun.com/exdb/mnist/, CIFAR10 https://www.cs.toronto.edu/~kriz/cifar.html, and Imagenette https://github.com/fastai/imagenette.

Code availability The codebase is publicly available at https://github.com/Cinofix/backdoor_learning_curves.

Declarations

Conflict of interest The authors declare that they have no Conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source,

provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Gu T, Liu K, Dolan-Gavitt B, Garg S (2019) Badnets: evaluating backdooring attacks on deep neural networks. IEEE Access 7:47230–47244
2. Liu Y, Ma S, Aafer Y, Lee W, Zhai J, Wang W, Zhang X (2018) Trojaning attack on neural networks. In: NDSS
3. Cinà AE, Grosse K, Demontis A, Vascon S, Zellinger W, Moser BA, Oprea A, Biggio B, Pelillo M, Roli F (2023) Wild patterns reloaded: a survey of machine learning security against training data poisoning. ACM Comput Surv 55(13s):1–39
4. Cinà AE, Grosse K, Demontis A, Biggio B, Roli F, Pelillo M (2024) Machine learning security against data poisoning: are we there yet? Computer 57(3):26–34
5. Cinà AE, Demontis A, Biggio B, Roli F, Pelillo M (2022) Energy-latency attacks via sponge poisoning
6. Chen X, Liu C, Li B, Lu K, Song D (2017) Targeted backdoor attacks on deep learning systems using data poisoning
7. Saha A, Subramanya A, Pirsiavash H (2020) Hidden trigger backdoor attacks. In: The thirty-fourth AAAI conference on artificial intelligence. AAAI, pp 11957–11965
8. Chen X, Salem A, Backes M, Ma S, Zhang Y (2020) Badnl: backdoor attacks against nlp models
9. Xi Z, Pang R, Ji S, Wang T (2021) Graph backdoor. In: 30th USENIX security symposium
10. Kiourti P, Wardega K, Jha S, Li W (2020) Trojdrl: evaluation of backdoor attacks on deep reinforcement learning. In: 2020 57th ACM/IEEE design automation conference (DAC). IEEE, pp 1–6
11. Cauwenberghs G, Poggio T (2001) Incremental and decremental support vector machine learning. In: NIPS, pp 409–415
12. Hastie T, Rosset S, Tibshirani R, Zhu J (2004) The entire regularization path for the support vector machine. JMLR 5:1391–1415
13. Koh PW, Liang P (2017) Understanding black-box predictions via influence functions. In: ICML
14. Pedregosa F (2016) Hyperparameter optimization with approximate gradient. In: Proceedings of the 33nd international conference on machine learning, ICML, vol 48, pp 737–746
15. Maclaurin D, Duvenaud D, Adams RP (2015) Gradient-based hyperparameter optimization through reversible learning. In: ICML, pp 2113–2122
16. Domke J (2012) Generic methods for optimization-based modeling. In: Proceedings of the fifteenth international conference on artificial intelligence and statistics, AISTATS, vol 22, pp 318–326
17. Zhang C, Bengio S, Singer Y (2022) Are all layers created equal? J Mach Learn Res 23(1):1–28
18. LeCun Y, Cortes C, Burges C (2010) Mnist handwritten digit database, vol 2. ATT Labs [Online]. http://yann.lecun.com/exdb/mnist
19. Krizhevsky A (2009) Learning multiple layers of features from tiny images. Technical report
20. Howard J. imagenette. https://github.com/fastai/imagenette/
21. Suya F, Mahloujifar S, Suri A, Evans D, Tian Y (2021) Model-targeted poisoning attacks with provable convergence. In: ICML. PMLR, pp 10000–10010
22. Donahue J, Jia Y, Vinyals O, Hoffman J, Zhang N, Tzeng E, Darrell T (2014) Decaf: a deep convolutional activation feature for generic visual recognition. In: ICML
23. Pasquale G, Ciliberto C, Odone F, Rosasco L, Natale L (2015) Teaching icub to recognize objects using deep convolutional neural networks. In: Proceedings of the 4th workshop on machine learning for interactive systems, MLIS, vol 43, pp 21–25
24. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks, vol 60, pp 84–90
25. Vabalas A, Gowen E, Poliakoff E, Casson AJ (2019) Machine learning algorithm validation with a limited sample size. PLoS ONE 14:e0224365
26. Tramèr F, Boneh D (2021) Differentially private learning needs better features (or much more data). In: ICLR
27. Dacrema MF, Cremonesi P, Jannach D (2019) Are we really making much progress? A worrying analysis of recent neural recommendation approaches
28. He K, Zhang X, Ren S, Sun J Deep residual learning for image recognition. In: 2016 IEEE conference on computer vision and pattern recognition, CVPR, pp 770–778
29. maintainers T contributors. TorchVision: PyTorch's Computer Vision Library
30. Mehta P, Bukov M, Wang C-H, Day AG, Richardson C, Fisher CK, Schwab D (2019) A high-bias, low-variance introduction to machine learning for physicists. Physics reports 810:1–124
31. Caruana R, Lawrence S, Giles CL (2000) Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In: NIPS, pp 402–408
32. Zhong H, Liao C, Squicciarini AC, Zhu S, Miller DJ (2020) Backdoor embedding in convolutional neural network models via invisibleperturbation. In: CODASPY, pp 97–108
33. Frederickson C, Moore M, Dawson G, Polikar R (2018) Attack strength vs. detectability dilemma in adversarial machine learning. In: 2018 int. joint conference on neural networks (IJCNN). IEEE, pp 1–8
34. Basu S, Pope P, Feizi S (2021) Influence functions in deep learning are fragile. In: ICLR
35. Bagdasaryan E, Shmatikov V (2023) Hyperparameter search is all you need for training-agnostic backdoor robustness. arXiv:2302.04977
36. Goldblum M, Tsipras D, Xie C, Chen X, Schwarzschild A, Song D, Madry A, Li B, Goldstein T (2020) Data security for machine learning: data poisoning, backdoor attacks, and defenses
37. Carnerero-Cano J, Munoz-González L, Spencer P, Lupu EC (2021) Regularization can help mitigate poisoning attacks... with the right hyperparameters. In: ICLR workshop on security and safety in machine learning system
38. Cinà AE, Vascon S, Demontis A, Biggio B, Roli F, Pelillo M (2021) The hammer and the nut: is bilevel optimization really needed to poison linear classifiers? In: IJCNN, pp 1–8
39. Baluta T, Shen S, Shinde S, Meel KS, Saxena P (2019) Quantitative verification of neural networks and its security applications. In: CCS
40. Lin J, Zhang A, Lécuyer M, Li J, Panda A, Sen S (2022) Measuring the effect of training data on deep learning predictions via randomized experiments. In: International conference on machine learning, ICML
41. Salem A, Wen R, Backes M, Ma S, Zhang Y (2022) Dynamic backdoor attacks against machine learning models. In: IEEE European symposium on security and privacy, EuroS &P 2022, pp 703–718
42. Severi G, Meyer J, Coull SE (2021) Explanation-guided backdoor poisoning attacks against malware classifiers. In: USENIX security symposium
43. Schwarzschild A, Goldblum M, Gupta A, Dickerson JP, Goldstein T (2021) Just how toxic is data poisoning? A unified benchmark

for backdoor and data poisoning attacks. In: International conference on learning representations (ICLR)
44. Li Y, Zhai T, Wu B, Jiang Y, Li Z, Xia S (2020) Rethinking the trigger of backdoor attack. arXiv:2004.04692
45. Frénay B, Verleysen M (2014) Classification in the presence of label noise: a survey. IEEE Trans Neural Netw Learn Syst 25:845–869
46. Teng CM (2000) Evaluating noise correction. In: Pacific rim international conference on artificial intelligence. https://api.semanticscholar.org/CorpusID:26238634
47. Teng CM (2001) A comparison of noise handling techniques. In: The Florida AI research society
48. Liu K, Dolan-Gavitt B, Garg S (2018) Fine-pruning: defending against backdooring attacks on deep neural networks. In: Int. symposium on research in attacks, intrusions, and defenses. Springer, p 273
49. Bajcsy P, Majurski M (2021) Baseline pruning-based approach to trojan detection in neural networks
50. Zeng Y, Qiu H, Guo S, Zhang T, Qiu M, Thuraisingham B (2020) Deepsweep: an evaluation framework for mitigating dnn backdoor attacks using data augmentation
51. Borgnia E, Cherepanova V, Fowl L, Ghiasi A, Geiping J, Goldblum M, Goldstein T, Gupta A (2021) Strong data augmentation sanitizes poisoning and backdoor attacks without an accuracy tradeoff. In: ICASSP. IEEE, pp 3855–3859
52. Hong S, Chandrasekaran V, Kaya Y, Dumitraş T, Papernot N (2020) On the effectiveness of mitigating data poisoning attacks with gradient shaping
53. Cook RD, Weisberg S (1980) Characterizations of an empirical influence function for detecting influential cases in regression. Technometrics 22(4):495–508
54. Christmann A, Steinwart I (2004) On robustness properties of convex risk minimization methods for pattern recognition. J Mach Learn Res 5:1007–1034
55. Geiping J, Fowl L, Somepalli G, Goldblum M, Moeller M, Goldstein T (2021) What doesn't kill you makes you robust (er): adversarial training against poisons and backdoors. arXiv:2102.13624
56. Li Y, Lyu X, Koren N, Lyu L, Li B, Ma X (2021) Anti-backdoor learning: training clean models on poisoned data, vol 34
57. Yang Y, Liu TY, Mirzasoleiman B (2022) Not all poisons are created equal: robust training against data poisoning. In: ICML, pp 25154–25165
58. Borgnia E, Geiping J, Cherepanova V, Fowl L, Gupta A, Ghiasi A, Huang F, Goldblum M, Goldstein T (2021) Dp-instahide: provably defusing poisoning and backdoor attacks with differentially private data augmentations
59. Du M, Jia R, Song D (2020) Robust anomaly detection and backdoor attack detection via differential privacy. In: ICLR

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.