Backdoor Learning Curves: Explaining Backdoor Poisoning Beyond Influence Functions

Antonio Emanuele Cinà
University of Venice
Venice, Italy
antonioemanuele.cina@unive.it

Kathrin Grosse
University of Cagliari
Cagliari, Italy
kathrin.grosse@unica.it

Sebastiano Vascon
University of Venice
Venice, Italy
sebastiano.vascon@unive.it

Ambra Demontis
University of Cagliari
Cagliari, Italy
ambra.demontis@unica.it

Battista Biggio
University of Cagliari
Cagliari, Italy
battista.biggio@unica.it

Fabio Roli
University of Cagliari
Cagliari, Italy
roli@unica.it

Marcello Pelillo
University of Venice
Venice, Italy
pelillo@unive.it

Abstract

Backdoor attacks inject poisoning samples during training, with the goal of enforcing a machine-learning model to output an attacker-chosen class when presented 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 success are not yet 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 success of backdoor attacks inherently depends on (i) the complexity of the learning algorithm, controlled by its hyperparameters, and (ii) the fraction of backdoor samples injected into the training set. These factors affect how fast a machine-learning model learns to correlate the presence of a backdoor trigger with the target class. Interestingly, our analysis shows that there exists a region in the hyperparameter space in which the accuracy on clean test samples is still high while backdoor attacks become ineffective, thereby suggesting novel criteria to improve existing defenses.

1 Introduction

Machine Learning (ML) has become pervasive in recent years and it is now a key component in several application domains, including self-driving cars and cyber-security. Its success is due to its ability to perform complex tasks, such as recognizing traffic signs or discriminating between legitimate applications and malware, with high accuracy. To learn to perform complex tasks accurately, ML necessitates a massive amount of data. Therefore, ML systems are usually based on models pre-trained or trained from scratch on data collected in the wild. The training process can thus be altered by malicious actors aiming to subvert the system's function, causing misclassification of a group of test samples. To this end, an attacker is able to perform an attack called *backdoor poisoning* [1–3]. It consists of injecting poisoning samples containing a particular pattern (called *trigger*) into the training data, with an attacker-chosen class label, to enforce the classifier to predict the attacker-chosen class whenever a sample containing that trigger is presented at test time.

Preprint. Under review.

Such backdoor attacks have been demonstrated in a plethora of scenarios, including publicly-available pre-trained models available for download [1, 2], data collected from untrusted sources [4], and training outsourced to an untrusted third party (e.g., a cloud provider) [1, 2]. Backdoors have also successfully been implanted into a variety of models, including vision [2] and language models [5], graph neural networks [6] and reinforcement learning [7]. Despite the high success of this attack, the factors affecting it remain not yet well understood.

In this work, we have analyzed the backdoor learning process to identify the main factors affecting the vulnerability of machine-learning models against this attack. For humans, the learning process is usually characterized 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 in this paper the notion of *backdoor learning curves* (Sect. 2). To generate these curves, we formulate backdoor learning as an incremental learning problem, and we assess how the loss on the backdoor samples decreases as they are gradually learned by the target model. The slope of this curve, which is connected to the notion of influence function, quantifies the ability of the model to learn a backdoor. Employing the proposed framework to study the backdoor learning process in different settings, we have obtained interesting insights and identified important factors that influence the vulnerability of learning algorithms against backdoor poisoning.

Our experimental analysis (Sect. 3) shows that factors that influence the backdoor poisoning success are the fraction of backdoor samples injected into the training data and the complexity of the target model, controlled via its hyperparameters. Surprisingly, our experimental findings reveal that there is a region in the hyperparameter space in which the model is highly accurate on the clean samples but remains robust to backdoor poisoning. The reason is that, to successfully learn a backdoor, the target model is required to increase its complexity, and this is not possible when models are sufficiently regularized. We believe that this observation may help identify novel criteria to improve existing defenses and inspire the development of new countermeasures.

To summarize, the main contributions of this work are:

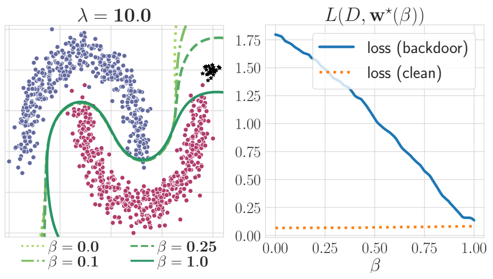
- we introduce *backdoor learning curves* as a powerful tool to thoroughly characterize the backdoor learning process;
- we introduce a metric, named *backdoor learning slope*, to quantify the ability of the classifier to learn backdoors;
- we identify two important factors that affect the vulnerability against backdoors;
- we unveil that there is a region in the hyperparameter space in which the classifiers are highly accurate and robust against backdoors, which may inspire 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.

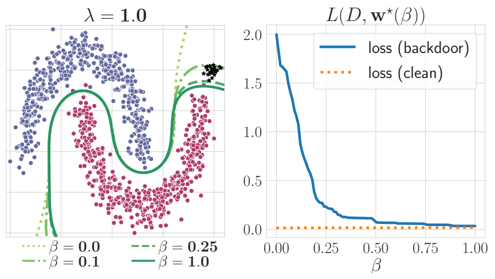
2 Backdoor Learning Curves

In this section, we introduce our framework to characterize backdoor poisoning by means of learning curves and their slope. Both the curves and their slope help us to understand how fast a classifier learns a backdoor trigger. To this end, we leverage previous work from incremental learning and study how *gradually* incorporating backdoor samples into the learning process affects the decision function learned by our ML model. We then present our approach to measure the impact of a backdoor on the learned parameters of the classifier. To this end, we propose to measure both the magnitude of difference in the parameters as well as the angular change of the decision function. To conclude the section, we discuss factors that influence the backdoor learning process and thus also affect the vulnerability of ML models against backdoor poisoning.

Backdoor Learning Curves. We start by introducing the notion of *backdoor learning curves*. To this end, let us first denote the input data and their labels respectively with $\boldsymbol{x} \in \mathbb{R}^d$ and $y \in \{1, .., c\}$, where c is 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 $\mathcal{P}_{\text{tr}} = (\hat{\boldsymbol{x}}_j, \hat{y}_j)_{j=1}^{m}$. The set of backdoor samples presented at test time is denoted with $\mathcal{P}_{\text{ts}} = (\hat{\boldsymbol{x}}_t, \hat{y}_t)_{t=1}^{k}$. $L(\mathcal{D}, \boldsymbol{w})$ is used to measure the loss attained by the classifier, parameterized by $\boldsymbol{w}$, on the dataset $\mathcal{D}$, while $\mathcal{L}(\mathcal{D}, \boldsymbol{w})$ additionally



(a) Strong regularization ($\lambda = 10$).



(b) Weak regularization ($\lambda = 1$).

Figure 1: Model complexity, backdoor learning curves and backdoor accuracy. Using an SVM with an RBF kernel ($\gamma = 10.0$) on a toy dataset in two dimensions, we show the influence of model complexity (as measured by λ, set to 10 (left plots) and 1 (right plots)). Each setting is visualized using two plots, where the left shows the data (dots) and decision surface for different β-values (green lines) and the right the backdoor learning curve. For the latter, the y-axis depicts the loss and the x-axis shows β. We plot both the backdoor loss (blue line) and the loss on clean data (orange dotted line). The slope of these curves represents the ease of the model to fit the backdoored samples: The model on the left struggles to fit the backdoors, and succeeds only with high β, whereas the model on the right already fits the model at low β.

includes any regularization term that may be optimized during training. Accordingly, the loss attained by the classifier on the backdoor samples presented at test time is given as

$$L(\mathcal{P}_{\text{ts}}, \boldsymbol{w}^\star(\beta)) = \sum_{t=1}^{k} \ell(\hat{\boldsymbol{x}}_t, \hat{y}_t, \boldsymbol{w}^\star(\beta)) \,, \tag{1}$$

where $\boldsymbol{w}^\star(\beta)$ represents the classifier parameters, obtained by solving the following learning problem:

$$\boldsymbol{w}^\star(\beta) \in \arg\min_{\boldsymbol{w}} \mathcal{L}(\mathcal{D}_{\text{tr}} \cup \mathcal{P}_{\text{tr}}, \boldsymbol{w}) = \sum_{i=1}^{n} \ell(\boldsymbol{x}_i, y_i, \boldsymbol{w}) + \beta \sum_{j=1}^{m} \ell(\hat{\boldsymbol{x}}_j, \hat{y}_j, \boldsymbol{w}) + \lambda \Omega(\boldsymbol{w}) \,. \tag{2}$$

In this learning problem, $\Omega(\boldsymbol{w})$ corresponds to the regularization term (e.g., $\|\boldsymbol{w}\|_2^2$), and λ is the associated regularization hyperparameter. We additionally introduce the hyperparameter $\beta \in [0, 1]$ in the learning problem to explicitly control the injection of the backdoor samples into the training process, inspired from previous work on incremental learning [8, 9]. As β ranges from 0 (unpoisoned classifier) to 1 (poisoned classifier), 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 and write $\boldsymbol{w}^\star(\beta)$.

We can now define the *backdoor learning curve* as the curve showing the behavior of the loss $L(\mathcal{P}_{\text{ts}}, \boldsymbol{w}^\star(\beta))$ in Eq. (1) as a function of β. Intuitively, the faster this curve decreases, the easier the target model is backdoored. We give an example of two backdoor learning curves under different regularization in Figure 1. The left plots depict a strongly regularized classifier. The corresponding backdoor learning curve shows that the classifier achieves low loss and high accuracy on the backdoor samples only after poisoning when $\beta = 1$. On the other hand, the classifier on the right is less regularized and more complex. As a consequence, the classifier learns to incorporate the backdoor samples much faster or at low β, highlighting that it might be more vulnerable to this attack.

Backdoor Learning Slope. To quantify this difference, or how fast an untainted classifier can be poisoned, is thus to compute the gradient of its backdoor learning curve at $\beta = 0$. In mathematical terms, we write

$$\frac{\partial L(\mathcal{P}_{\text{ts}}, \boldsymbol{w}^\star(\beta))}{\partial \beta} = \frac{\partial L}{\partial \boldsymbol{w}} \frac{\partial \boldsymbol{w}^\star}{\partial \beta} \,. \tag{3}$$

While the first term is straightforward to compute, 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 from previous work in incremental learning [8], we assume that, while increasing β, the solution maintains the optimality (Karush-Kuhn-Tucker, KKT) conditions intact. This equilibrium implies that $\nabla_\beta \nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}^\star) + \frac{\partial \boldsymbol{w}^\star}{\partial \beta} \nabla^2_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}^\star) = \mathbf{0}$. Based on this condition, we obtain the derivative of interest,

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

and the complete gradient as

$$\frac{\partial L(\mathcal{P}_{\text{ts}}, \boldsymbol{w}^\star(\beta))}{\partial \beta} = -\nabla_{\boldsymbol{w}} L \cdot (\nabla^2_{\boldsymbol{w}} \mathcal{L})^{-1} \cdot \nabla_\beta \nabla_{\boldsymbol{w}} \mathcal{L}\,. \tag{5}$$

This gradient 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 and Liang [10]. Koh and Liang propose to measure how relevant a training point is for the predictions of a given test point by computing $\frac{\partial L}{\partial \beta} = \sum_t \sum_j \mathcal{I}_{\text{up,loss}}(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{x}}_j)$. To understand how this gradient can be efficiently computed via Hessian-vector products and other approximations, we refer the reader to [10] as well as to recent developments in gradient-based bilevel optimization [11–15].

The main differences between our and Koh and Liang's [10] approach stem 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, using the gradient of the loss wrt. β directly comes with two disadvantages. First, the slope is inverse to β and second, to obtain comparable results across classifiers, we need to rescale the slope. We thus transform the gradient as

$$\theta = -\frac{2}{\pi} \arctan\left(\left. \frac{\partial L}{\partial \beta} \right|_{\beta=0} \right) \in [-1, 1]\,, \tag{6}$$

where we use the negative sign to have positive values correlated with faster backdoor learning (i.e., the loss is decreasing faster as β grows). Computing $2/\pi$ of the gradient allows us to rescale the slope to be in the interval between $[-1, 1]$.

Backdoor Impact on Learning Parameters. After introducing the previous plot and measure, we are able to quantify how backdoors are learned by the model. However, we are also interested in whether the model increases in complexity when learning the backdoor in addition to the original task. We thus propose to monitor how the classifier parameters deviate from their initial, unbackdoored values once a backdoor is added. To this end, 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,

$$\rho = \|\boldsymbol{w}_\beta\| \in [0, \infty), \text{ and } \nu = \frac{1}{2}\left(1 - \frac{\boldsymbol{w}_0^\top \boldsymbol{w}_\beta}{\|\boldsymbol{w}_0\| \|\boldsymbol{w}_\beta\|}\right) \in [0, 1]\,. \tag{7}$$

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

Remarks. Before we turn to our experiments, we would like to discuss some properties of the introduced concepts. It is worth noting that not only β influences backdoor success, but also the amount of clean and backdoored input samples as well as input dimensionality, the complexity of the clean task, the size of the trigger, and the level of regularization.

We would also like to point out that the clean accuracy remains largely unaffected as β increases. The intuition behind this is that the classifier is able to incorporate the backdoor without changing the benign classification output, a result that has also been reported by prior works [1–3]. When heavy regularization is applied, however, clean accuracy might be affected. We then also expect this regularization to have an influence on the backdoor slope as visualized in Figure 1. More concretely, if regularization affects the backdoor slope to a value close to 0, we expect almost no change in the classifier parameters for $\beta > 0$. In this case, the classifier is very robust against backdoor poisoning.

3 Experiments

Using the previous methodology, we carried out an empirical analysis on linear and nonlinear classifiers under different regularization. The goal of our analysis is to show the properties and components that affect the slope of the backdoor learning curve and thus the sensitivity of a model to backdoor attacks.

Our experiments show that a key factor affecting the backdoor slope and thereby backdoor accuracy is the regularization term λ. We find evidence that there exists an area where the accuracy on benign samples remains high, yet the classifier is robust to the backdoor. Subsequently, our experiments show that increasing the fraction of injected poisoning points brings the classifier to learn the backdoor faster. This occurs without sacrificing clean test accuracy as long as the classifier has enough flexibility. Finally, we analyze the change of the parameters weights during the backdoor learning process. We found that linear classifiers have to increase their complexity in order to learn the backdoor. Conversely, nonlinear ones are already flexible enough to learn the backdoor without significantly increase their complexity.

In the following, we first detail the considered experimental setup, then we report the results and highlight the most interesting finding.

3.1 Experimental Setup

To describe the setup of our experiments, we first detail the datasets and then describe models, the implemented backdoor attacks and the implementation.

Datasets. In our analysis we consider the MNIST [16] and CIFAR10 [17] datasets. Similarly to [3], we focus on two-class subproblems. On MNIST, we chose $7 \text{ vs } 1$, $3 \text{ vs } 0$, and $5 \text{ vs } 2$, as on these pairs our models exhibited the highest clean accuracy. On CIFAR10, analogous to prior work [3], we chose *airplane vs frog*, *bird vs dog*, and *airplane vs truck*. In the following section, we focus on the results of one pair (7 vs 1 on MNIST, and *airplane vs frog* on CIFAR10), as the results of the other pairs are analogous (as can be confirmed in the Appendix).

Models and Training phase. We consider various models to thoroughly analyze how learning a backdoor affects a model. More specifically, we analyse Support Vector Machines (SVMs), Logistic Regression (LC), Ridge Classifiers (RCs), and SVM with Radial Basis Kernel (RBF). We train all models directly on the raw data or in a transfer learning setting. In the latter, we take a pre-trained Alexnet [18] used as a feature extractor, analogous to the setting by Saha et al. [3]. This setting is used for the more challenging CIFAR10 dataset. On MNIST, we train the classifiers directly on the pixel values scaled in the range $[0, 1]$.

Hyperparameters. Our goal is to study which components affect backdoor learning curves. As regularization is a significant factor, we decide to analyze the backdoor learning process for a wide range of regularizing hyperparameters. More specifically, we vary λ in $\{1\text{e}{-}03, 1\text{e}{-}02, 1\text{e}{-}01, 1, 1\text{e}{+}01, 1\text{e}{+}02, 1\text{e}{+}03, 1\text{e}{+}04\}$. Concerning the RBF kernel, we let γ take uniformly spaced values on a log scale in $[5\text{e}{-}04, 5\text{e}{-}02]$ for MNIST and $[1\text{e}{-}04, 1\text{e}{-}02]$ for CIFAR. Furthermore, we chose $\lambda \in \{1\text{e}{-}03, 1\text{e}{-}02, 1\text{e}{-}01, 1, 1\text{e}{+}01\}$ on the RBF kernel. For all hyperparameters, the accuracy on clean data (e.g., without backdoors) is $> 90\%$ for MNIST and CIFAR10. An exception is the configuration with $\gamma = 5\text{e}{-}02$ and $\lambda = 1\text{e}{+}01$, where the accuracy on samples without trigger is only 75%.

Backdoor Attacks. On both datasets, we add a trigger to 10% of the training data if not stated otherwise. The trigger for MNIST is a random 3×3 patch. On CIFAR10, we increase the size to 8×8 to strengthen the attack [3]. We add the trigger pattern in the lower right corner of the image [2]. Samples from MNIST and CIFAR10 with and without trigger can be found in Figure 5. After training, the presence of a trigger forces the backdoored model to predict the i-th class as class $(i + 1)\%2$ [2]. However, in contrast to previous approaches [2], 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. By using different trigger patterns, we improve the effectiveness of the attack.

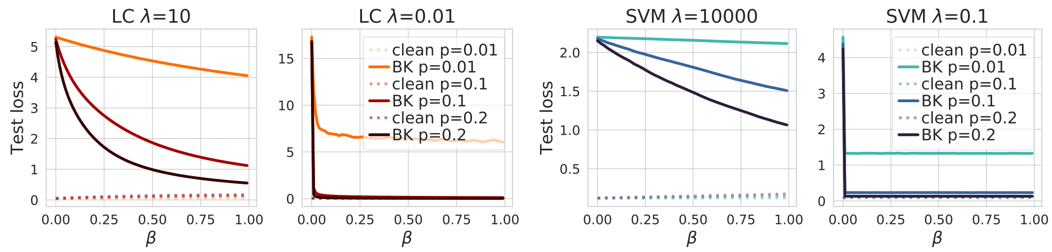


Figure 2: Backdoor learning curves for the logistic classifier (LC) on MNIST 7-1 with $\lambda \in \{10, 0.01\}$ *(left)*, and for SVM on CIFAR10 *airplane vs frog* with $\lambda \in \{10000, 0.1\}$ *(right)*. Darker lines represent a higher fraction of poisoning samples p injected into the training set. We report the loss on backdoored (BK) test (solid line), and the loss for clean test samples (dashed line).

Implementation. We rely on public libraries to implement our models [19, 20], standard procedures like reading and accessing data [21], mathematical computations like for example norms or gradients [22, 23] and to create the plots in our paper [24, 25].

3.2 Experimental Results

Having detailed the experimental setup, we can now discuss our experimental results.

Backdoor Learning Curves. In this experiment, we investigate how the amount of backdoored samples in the training set affects backdoor learning of differently regularized classifiers. In Figure 2 we show the backdoor learning curves under different values of β. We report the results of LC trained on MNIST (left) and of SVM on CIFAR10 (right), full results are in the Appendix. We increase the fraction of poisoning samples $p \in \{0.01, 0.1, 0.2\}$, where larger values of p correspond to darker lines in the plot.

As visible in Figure 2, both smaller λ and larger p increase the slope of the backdoor learning curve. More specifically, when decreasing λ, the backdoor learning curve drops faster. This drop suggests that more complex classifiers are able to learn the additional backdoor classification task at lower β. In addition, the fraction of poisoning samples p injected into the training set plays an important role. As p increases, the backdoor learning curves get steeper, and consequently the classifier learns the backdoor at lower β. Furthermore, with larger p, the accuracy for benign data does not change. We conclude that less regularized, and thus more complex models indeed learn the backdoor faster than their regularized counterparts. However, the amount of backdoored samples in the training set also influences how well a backdoor is learned, and how accurate the classifier is on data without trigger.

Backdoor Slope. As seen in Section 2, the influence function proposed by Koh and Liang [10] is the partial derivative of the backdoor learning curve at the point $\beta = 0$. We investigate the backdoor effectiveness through the lens of the influence functions when the fraction of poisoning at $p = 0.1$We train the target classifiers on the poisoned data under the previously defined regularization conditions. Figure 3 shows the relationship between the backdoor slope and the backdoor effectiveness. We measured the effectiveness as the percentage of samples with trigger that mislead the classifier. For the SVM with RBF kernel we report results for $\gamma \in \{5\text{e}{-04}, 5\text{e}{-03}\}$ (MNIST) and $\gamma \in \{1\text{e}{-04}, 1\text{e}{-03}\}$ (CIFAR10).

As visible in Figure 3, for both MNIST (top) and CIFAR10 (bottom), there is a clear relationship between the backdoor slope and its effectiveness. More concretely, higher backdoor slope always implies a higher accuracy of the model on samples with trigger. Higher slope thus implies higher vulnerability towards backdoors. Intriguingly, the plots show a region where the accuracy of the classifier on benign samples is high (red triangles), yet the classifier exhibits low accuracy on samples with trigger (blue x). Similar observations hold for the SVM with an RBF kernel. In addition to the λ-parameter, we must also consider the RBF parameter γ. As visible in the rightmost plots in Figure 3, the best trade-off is achieved with low λ and smaller γ. Small values of γ constrain the classifier too heavily. As a consequence, the classifier is not flexible enough to learn the backdoor task in addition to the original task. As our previous experiments showed that the slope of the backdoor is affected by the regularization term λ, we conclude that regularization enables us to configure a

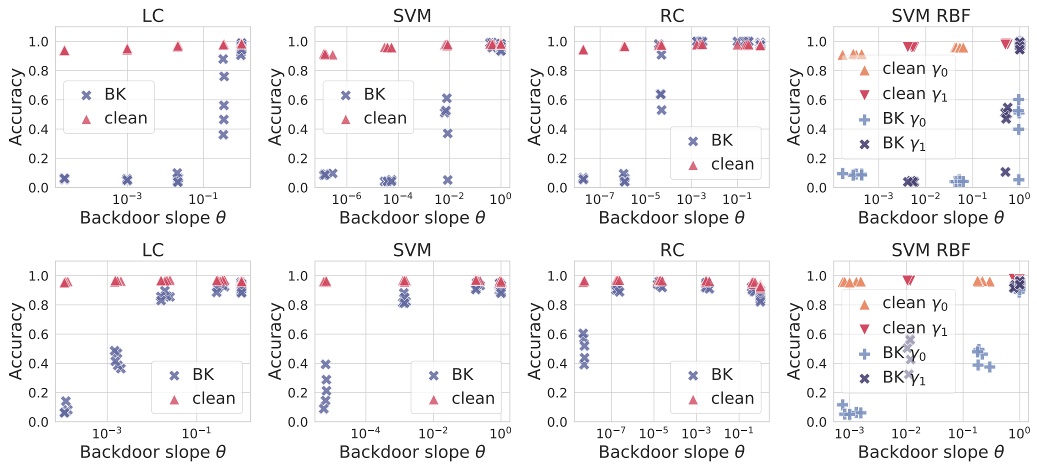


Figure 3: Backdoor slope θ vs backdoor (BK) effectiveness (blue) and clean accuracy (red) on MNIST 7-1 (top) and CIFAR10 *airplane vs frog* (bottom). We chose the γ parameter for the RBF kernel as $\gamma_0 = 5\text{e}{-04}$ (orange triangle for clean data, light blue plus for data with trigger) and $\gamma_1 = 5\text{e}{-03}$ (red inverted triangle for clean data, dark blue x for data with trigger) for MNIST. For CIFAR10, we set $\gamma_0 = 1\text{e}{-04}$ (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).

classifier that learns the original task but not the backdoor. In other words, there is a trade-off between the accuracy on the original task and robustness to backdoor classification.

Empirical Parameter Deviation Plots. After having investigated which factors influence backdoor effectiveness, we now turn to the question whether a model that contains a backdoor is more complex than its unbackdoored counterpart. To this end, we investigate the two measures proposed in Section 2, ρ and ν. 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 in Figure 4. The fraction of poisoning points is $p = 0.1$, or 10%, for both MNIST (top) and CIFAR10 (bottom).

On linear classifiers, $\rho(\boldsymbol{w})$ increases during the backdoor learning process. This is equivalent to an increase of 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. Here, when the regularization term increases, there is no need for the classifier to increase its weights significantly. We conclude 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.

3.3 Explaining Backdoor Predictions

In the following we propose a first attempt to interpret the decision function of the poisoned classifiers. In particular, given a sample x we compute the gradient of the classifier's decision function with respect to x. We use a SVM with regularization $\lambda = 1\text{e}{-02}$ for MNIST 6-0 and CIFAR10 *airplane vs frog*, and we report the results in Figure 5.

For MNIST we consider a digit 7 with the trigger (left) and we show the gradient of the clean classifier's decision function. We report the outcome of the gradient from the clean (middle) and poisoned (right) classifiers for the corresponding input. We remark that since we train a linear classifier on the input space, then the derivative coincide with the classifier's weights. Intriguingly, the classifier's weights increased their magnitude and now has high values on the bottom right corner, where the trigger is located.

From CIFAR10, we show a poisoned airplane (left). We report the gradient mask obtained by considering the maximum value for each channel, both for the clean (middle) and backdoored (right) classifier. Also in this case, the backdoored model shows higher values in the bottom right region,

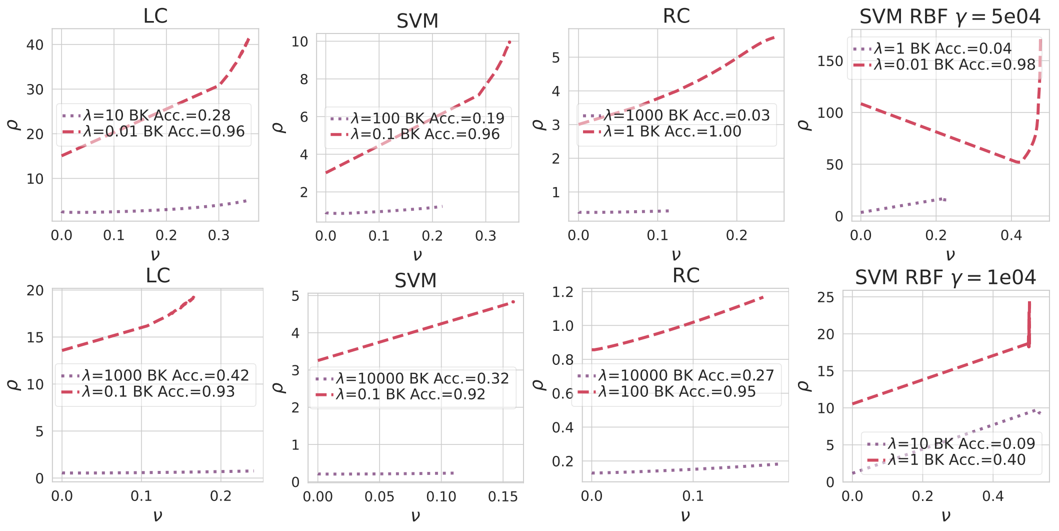


Figure 4: Backdoor weight deviation for the logistic classifier (LC), SVM, the ridge classifier (RC) and SVM with RBF kernel on MNIST (top) and CIFAR10 (bottom). We specify regularization parameter λ and backdoor (BK) accuracy for each setting in the legend of each plot.

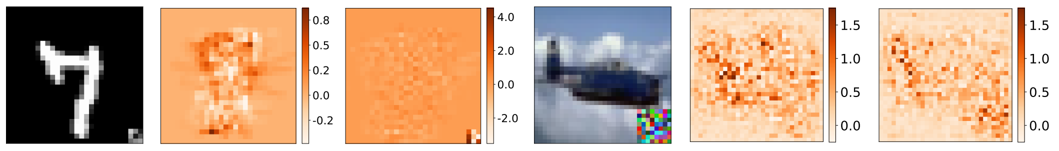


Figure 5: The first half of the plots consider the case of MNIST 7-1, and the second half is for CIFAR10 *airplane vs frog*. For each block we have: the poisoned test sample under consideration (left); the gradient of the untainted SVM decision's boundary with respect to the input (middle); the gradient of the poisoned SVM decision's boundary with respect to the input (right). For CIFAR10 we consider the maximum gradient of each pixel among the 3 channels.

corresponding to the trigger location. This means that the analyzed classifiers assigns high importance to the trigger in order to discriminate the class of the input points. Summarizing, the plots in Figure 5 confirm our results from Section 3.2 regarding the increase in complexity due to the backdoor.

Figure 6 shows the corresponding top 7 most influential samples in the poisoned training set. We plot on the left the poisoned input sample (with a red border) and show next to it the points which yield the highest influence function values. For both MNIST (top) and CIFAR10 (bottom), among the most influential training points are the ones with the trigger.

4 Related Work and Relation of Backdoor Learning Curves to Existing Defenses

We first review general related work in the area of backdoors. Afterwards, we discuss specific defenses that are related to our study of regularization and backdoor effectiveness. We conclude the section by discussing related work in the area of influence functions.

Backdoors Although introduced recently [2, 26], there is a plethora of backdoor attacks and backdoor defenses. An overview can for example be found in the survey by Gao et al [27] or Goldblum et al. [28]. Although many defenses have been proposed, there is an ongoing arms-race [29, 30]. Few works study backdoors not as attack or defense but rather as a phenomenon. For example, Frederickson et al. [31] study the trade-off between strength of an attack, effectiveness and detectability in backdoor triggers. We instead focus on the relationship between model complexity or regularization and attack effectiveness. Wang et al. [32] study the vulnerability of ensembles versus individual classifiers. However, we focus on the robustness or vulnerability of individual classifiers. Carnerero-Cano

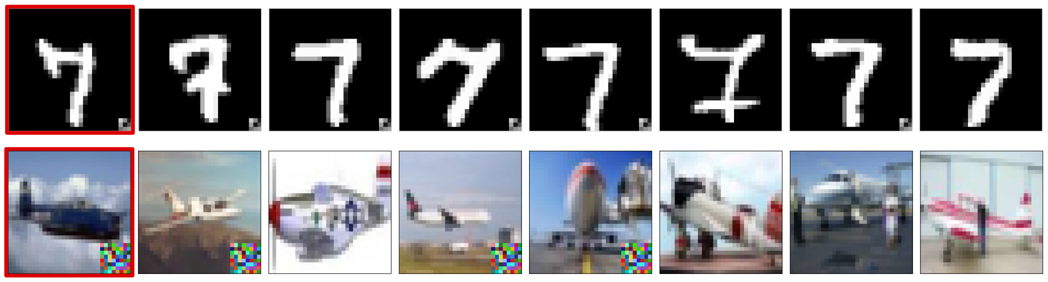


Figure 6: We report the top 7 most influent samples from the poisoned SVM on the prediction of the samples with the red border.

et al. [33] study the effect regularization has on poisoning, however focusing on indiscriminate attacks that reduce accuracy (e.g., not on backdoors). Finally, Baluta et al. [34] study backdoor generalization using their formal framework in binary neural networks. They conclude that the trigger is only effective when combined with images from the training distribution. We do not study out of distribution samples with a trigger, and instead focus on the effect of regularization on backdoors.

Our work highlights an intriguing relationship between model complexity and backdoor effectiveness. Although we are the first to experimentally show and outline this relationship, several approaches have used this relationship indirectly to **defend backdoors**. For example, data augmentation can be considered as regularization [35]. Consequently, approaches relying on data augmentation indirectly implement our approach [36, 37]. Furthermore, also pruning approaches affect model complexity and are thus loosely related to our findings [38, 39]. Last but not least, [40] show that gradient shaping, or clipping the gradients during training, greatly mitigates the effect of backdoor attacks. As a limit on the gradient's size results in a limit in the weight update, this approach is also loosely related. Of these approaches, [37] report that they are able to defeat the attack without a decrease in clean accuracy, a finding backed up in our work.

Influence functions stem from robust statistics [41] and were later used as a tool to measure the influence of training points on the classification output on for example an SVM [42] or deep learning [10]. However, Basu et al. [43] showed later that influence functions are more reliable on shallow models. More specifically, influence functions are strongly affected by weight decay and the choice of the test point the influence is computed for.

5 Conclusions, Limitations, Future Work and Impact

In this work, we have devised a framework that allows analyzing the factors that influence the success of backdoor poisoning. Thanks to the proposed mathematical tools, we found that the more relevant factors are the number of backdoor samples injected into the training dataset and the target model's complexity. Furthermore, we have discovered that there exists a region in the hyperparameter space in which the accuracy on clean test samples is still high while backdoor attacks become ineffective.

Our experiments were carried out on convex learners. It is thus an open question whether our findings generalize to deeper models. Basu et al. [43] show, influence functions, a concept closely related to our backdoor learning curves, tend to be unstable on deep learning models. We thus worked in a transfer learning setting to overcome these difficulties. Moreover, measuring the complexity of deep learning models is an open research question. Hence, we leave the interpretation and investigation of the found trade-off between clean and backdoor accuracy in deep learning models for future work. Our framework does also not encompass all possible factors influencing backdoor effectiveness, only β, the amount of backdoored samples, and regularization. Other factors like the amount of clean and poisoned data, the size and shape of the trigger, or the size of the input space are left out. We leave the integration of all these factors to future work, although they could be seamlessly analyzed with our framework.

Finally, we briefly comment on the potential negative impact of our work on society. Our work increases the understanding of a severe security threat against AI systems, and we cannot exclude this knowledge will be exploited by malicious players to improve their attack. Nevertheless, we strongly believe this knowledge can help pave the way towards designing ML systems more robust against backdoor poisoning attacks.

Acknowledgments and Disclosure of Funding

The research reported in this paper has been partly funded by BMK, BMDW, and the Province of Upper Austria in the frame of the COMET Programme managed by FFG in the COMET Module S3AI.

This work has also been partially supported by the PRIN 2017 RexLearn project (grant no. 2017TWNMH2), funded by the Italian Ministry of Education, University and Research; and by BMK, BMDW and the Province of Upper Austria in the within the COMET programme managed by FFG in the COMET S3AI module.

References

[1] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *NDSS*, 2018.

[2] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019.

[3] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. In *AAAI*, 2020.

[4] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *NeurIPS*, 2018.

[5] Xiaoyi Chen, Ahmed Salem, Michael Backes, Shiqing Ma, and Yang Zhang. Badnl: Backdoor attacks against nlp models. *arXiv preprint arXiv:2006.01043*, 2020.

[6] Zhaohan Xi, Ren Pang, Shouling Ji, and Ting Wang. Graph backdoor. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*, 2021.

[7] Panagiota Kiourti, Kacper Wardega, Susmit Jha, and Wenchao Li. Trojdrl: evaluation of backdoor attacks on deep reinforcement learning. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2020.

[8] Gert Cauwenberghs and Tomaso Poggio. Incremental and decremental support vector machine learning. *Advances in neural information processing systems*, pages 409–415, 2001.

[9] Trevor Hastie, Saharon Rosset, Robert Tibshirani, and Ji Zhu. The entire regularization path for the support vector machine. *J. Mach. Learn. Res.*, 5:1391–1415, 2004.

[10] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *ICML*, 2017.

[11] Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1540–1552. PMLR, 26–28 Aug 2020.

[12] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In Jennifer Dy and Andreas Krause, editors, *35th ICML*, volume 80 of *Proc. Mach. Learn. Res.*, pages 1568–1577. PMLR, 2018.

[13] F. Pedregosa. Hyperparameter optimization with approximate gradient. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 737–746, New York, New York, USA, 20–22 Jun 2016. PMLR.

[14] Dougal Maclaurin, David Duvenaud, and Ryan P. Adams. Gradient-based hyperparameter optimization through reversible learning. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 2113–2122. JMLR.org, 2015.

[15] Justin Domke. Generic methods for optimization-based modeling. In Neil D. Lawrence and Mark Girolami, editors, *15th Int'l Conf. Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pages 318–326, La Palma, Canary Islands, 21–23 Apr 2012. PMLR.

[16] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[17] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114, 2012.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[20] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[21] The pandas development team. pandas-dev/pandas: Pandas, February 2020. URL `https://doi.org/10.5281/zenodo.3509134`.

[22] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[23] Marco Melis, Ambra Demontis, Maura Pintor, Angelo Sotgiu, and Battista Biggio. secml: A python library for secure and explainable machine learning. *arXiv preprint arXiv:1912.10013*, 2019.

[24] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9 (3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

[25] Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021. doi: 10.21105/joss.03021. URL `https://doi.org/10.21105/joss.03021`.

[26] Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

[27] Yansong Gao, Bao Gia Doan, Zhi Zhang, Siqi Ma, Jiliang Zhang, Anmin Fu, Surya Nepal, and Hyoungshick Kim. Backdoor attacks and countermeasures on deep learning: A comprehensive review. *CoRR*, abs/2007.10760, 2020. URL `https://arxiv.org/abs/2007.10760`.

[28] Micah Goldblum, Dimitris Tsipras, Chulin Xie, Xinyun Chen, Avi Schwarzschild, Dawn Song, Aleksander Madry, Bo Li, and Tom Goldstein. Data security for machine learning: Data poisoning, backdoor attacks, and defenses. *arXiv preprint arXiv:2012.10544*, 2020.

[29] Te Juin Lester Tan Tan and Reza Shokri. Bypassing backdoor detection algorithms in deep learning. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2020.

[30] Anh Nguyen and Anh Tran. Wanet–imperceptible warping-based backdoor attack. In *ICLR*, 2021.

[31] Christopher Frederickson, Michael Moore, Glenn Dawson, and Robi Polikar. Attack strength vs. detectability dilemma in adversarial machine learning. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[32] Shir Li Wang, Kamran Shafi, Chris Lokan, and Hussein A Abbass. Robustness of neural ensembles against targeted and random adversarial learning. In *International Conference on Fuzzy Systems*, pages 1–8. IEEE, 2010.

[33] Javier Carnerero-Cano, Luis Munoz-González, Phillippa Spencer, and Emil C Lupu. Regularization can help mitigate poisoning attacks... with the right hyperparameters. In *ICLR Workshop on Security and Safety in Machine Learning System*, 2021.

[34] Teodora Baluta, Shiqi Shen, Shweta Shinde, Kuldeep S Meel, and Prateek Saxena. Quantitative verification of neural networks and its security applications. In *CCS*, 2019.

[35] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[36] Yi Zeng, Han Qiu, Shangwei Guo, Tianwei Zhang, Meikang Qiu, and Bhavani Thuraisingham. Deepsweep: An evaluation framework for mitigating dnn backdoor attacks using data augmentation. *arXiv preprint arXiv:2012.07006*, 2020.

[37] Eitan Borgnia, Valeriia Cherepanova, Liam Fowl, Amin Ghiasi, Jonas Geiping, Micah Goldblum, Tom Goldstein, and Arjun Gupta. Strong data augmentation sanitizes poisoning and backdoor attacks without an accuracy tradeoff. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3855–3859. IEEE, 2021.

[38] Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 273–294. Springer, 2018.

[39] Peter Bajcsy and Michael Majurski. Baseline pruning-based approach to trojan detection in neural networks. *arXiv preprint arXiv:2101.12016*, 2021.

[40] Sanghyun Hong, Varun Chandrasekaran, Yiğitcan Kaya, Tudor Dumitraş, and Nicolas Papernot. On the effectiveness of mitigating data poisoning attacks with gradient shaping. *arXiv preprint arXiv:2002.11497*, 2020.

[41] R Dennis Cook and Sanford Weisberg. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*, 22(4):495–508, 1980.

[42] Andreas Christmann and Ingo Steinwart. On robustness properties of convex risk minimization methods for pattern recognition. *The Journal of Machine Learning Research*, 5:1007–1034, 2004.

[43] Samyadeep Basu, Philip Pope, and Soheil Feizi. Influence functions in deep learning are fragile. In *ICLR*, 2021.

A Appendix

A.1 Additional Experimental Results

In this Appendix, we present additional experimental results that confirm our claims. Firstly, whereas in the paper, we have shown the backdoor learning curves only for two classifiers, here, we report them for all the classifiers considered in this work. In particular, here we consider:

- Logistic Classifier (LC) with $\lambda \in \{10, 0.01\}$ (MNIST) or $\lambda \in \{1000, 0.1\}$ (CIFAR10),
- Support Vector Machine (SVM) with $\lambda \in \{100, 0.1\}$ (MNIST) or $\lambda \in \{10000, 0.1\}$ (CIFAR10),
- Ridge Classifier (RC) with $\lambda \in \{1000, 1\}$ (MNIST) or $\lambda \in \{10000, 100\}$ (CIFAR10),
- SVM with an RBF kernel, where $\lambda \in \{1, 0.01\}$ and $\gamma = 5\text{e}{-}03$ (MNIST) or $\lambda \in \{10, 1\}$ and $\gamma = 1\text{e}{-}03$ (CIFAR10).

Moreover, we compare the results obtained on the class pairs considered in the paper (7 vs 1 on MNIST and *airplane vs frog* on CIFAR10) with the one obtained on different pairs. As other pairs, we consider 3 vs 0 and 5 vs 2 on MNIST and *bird vs dog* and *airplane vs truck* on CIFAR10.

Backdoor Learning Curves. In Figure 7 and 10, we plot the backdoor learning curves obtained on the same class pair considered in the paper (7 vs 1 on MNIST and *airplane vs frog* on CIFAR10).

In the following, we present the results obtained on class pairs different from those considered in the paper. We show the results obtained on the class pairs 3 vs 0 (5 vs 2) of the MNIST dataset in Figure 8 (Figure 9). We further show the results obtained on the class pairs *bird vs dog* (*airplane vs truck*) of the CIFAR10 dataset in Figure 11 (Figure 12).

These additional experimental results are coherent with the ones reported in the paper.

Backdoor Slope. In Figure 13 and 14, we report the backdoor learning slope, computed with $p = 0.1$, for all the considered classifiers and all subset pairs. For the SVM-RBF, we have reported the results obtained on MNIST with $\gamma_0 = 5\text{e}{-}04$ and $\gamma_1 = 5\text{e}{-}03$ and on CIFAR10 with $\gamma_0 = 5\text{e}{-}04$ and $\gamma_1 = 5\text{e}{-}03$. The results do not show significant variation with respect to the ones reported in the paper.

Empirical Parameter Deviation Plots. In Figure 15 and Figure 16, we plot the results of the empirical parameter deviation analysis on all computed with $p = 0.1$ for all the considered classifiers and class pairs. As before, 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.

A.1.1 The Influence of γ on SVM RBF Backdoor vulnerability

In the paper, we studied the influence of λ on the vulnerability to backdoors. When the classifier is an SVM with an RBF kernel, the kernel's parameter γ plays a role too. In the paper experiments, we kept the value of this parameter fixed, and here, we study its influence on the classifier vulnerability against backdoors.

In Figure 17 we report the accuracies considering different combinations of γ and λ for the class pair 7 vs 1 of the MNIST dataset (upper row) and for the class pair *airplane vs frog* of the CIFAR10 dataset, on the clean (left) and backdoored (right) test dataset.

The clean accuracy behaves differently depending on the dataset. On MNIST, the accuracy increases when λ increases and γ decreases. Whereas on CIFAR10, it decreases slightly when λ assumes high values. The accuracy on the backdoored dataset, independently from the dataset, is lower for high λ and small γ. High values of γ make the accuracy on the backdoored samples decrease since the model starts to overfit the backdoor samples, failing to generalize well for other test samples with the trigger. However, for large gamma also the accuracy on the clean dataset decreases due to overfitting. The outcome desired by the defender is to have high accuracy on the clean dataset and low accuracy on the backdoored dataset. This figure confirms that having small γ and high λ offer a good trade-off between these two accuracies.

To gain more insights about the robustness caused by high values of γ, we have analyzed the backdoor learning curves, and the classifier's parameters change due to the backdoor learning. In particular, we have considered an SVM-RBF with $\gamma_0 = 5\mathrm{e}{-04}$ and $\gamma_1 = 5\mathrm{e}{-03}$ trained on the class pair 7 vs 1 of the MNIST dataset and one with $\gamma_0 = 1\mathrm{e}{-04}$ and $\gamma_1 = 1\mathrm{e}{-03}$ trained on the class pair *airplane vs frog* of the CIFAR10 dataset. We have reported the learning curves in Figure 18. On both the dataset, when decreasing γ, the backdoor learning curves are flatter, and the test loss increases, suggesting higher robustness. Furthermore, we have reported the parameter deviation plots in Figure 19. This figure shows that, as γ increases, 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, it needs to increase its complexity to learn the backdoor. Overall, those experiments confirm that a classifier, to learn a backdoor, has to increase its complexity (if it is not already highly complex). This is not possible for highly regularized classifiers, which are, therefore, preferable.

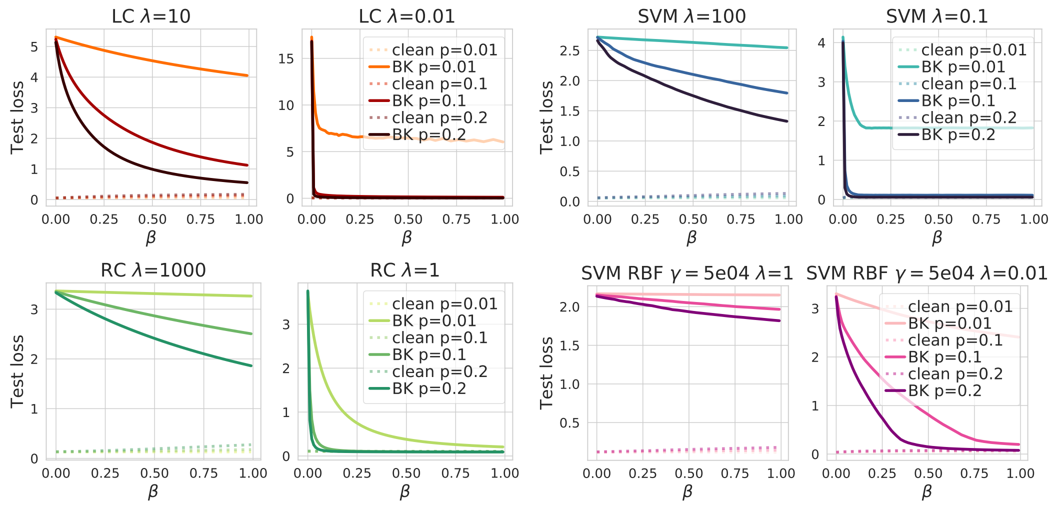


Figure 7: Backdoor learning curves for different classifiers trained on MNIST 7-1. Darker lines represent a higher fraction of poisoning samples p injected into the training set. We report with solid (dashed) lines the loss on the backdoored (clean) test dataset.

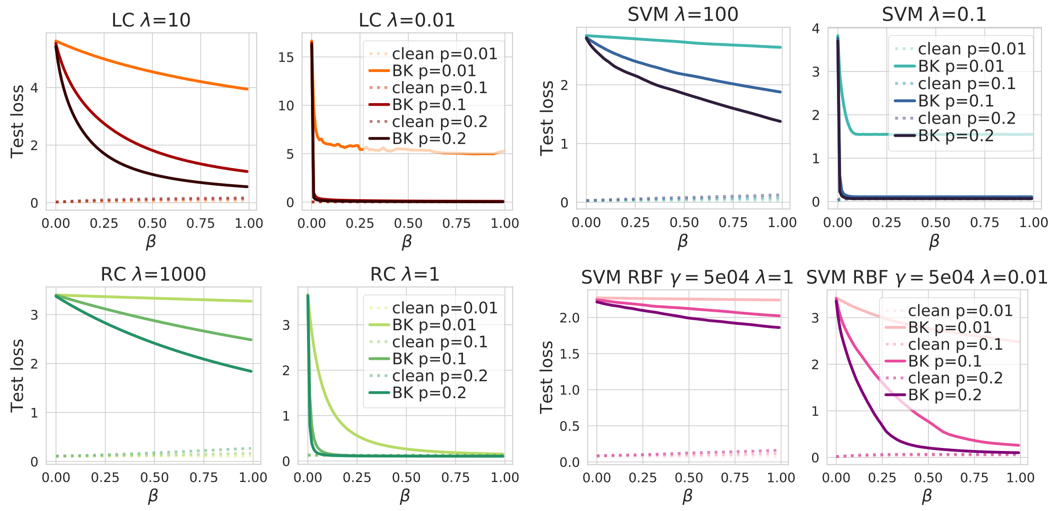


Figure 8: Backdoor learning curves for different classifiers trained on MNIST 3-0. See the caption of Figure 8 for further details.

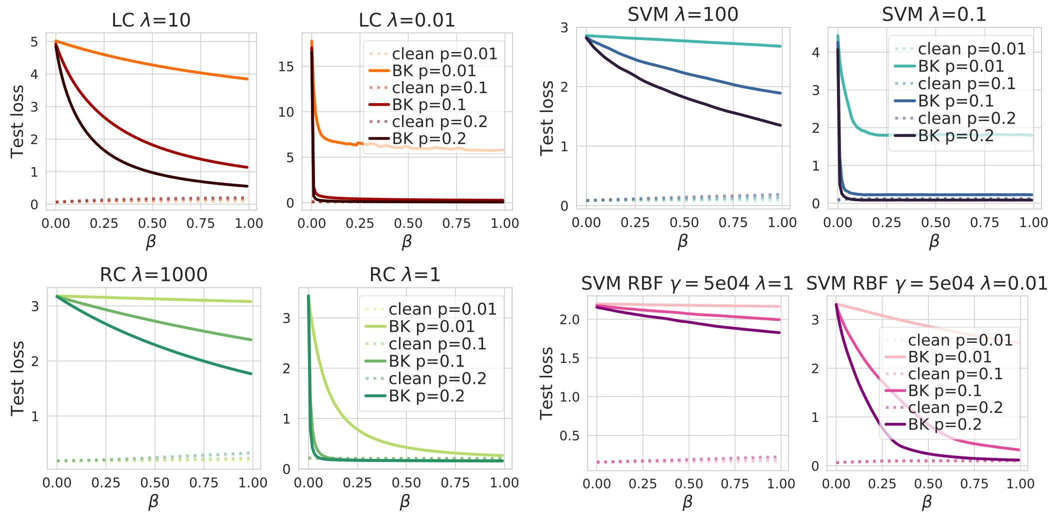


Figure 9: Backdoor learning curves for different classifiers trained on MNIST 5-2. See the caption of Figure 9 for further details.

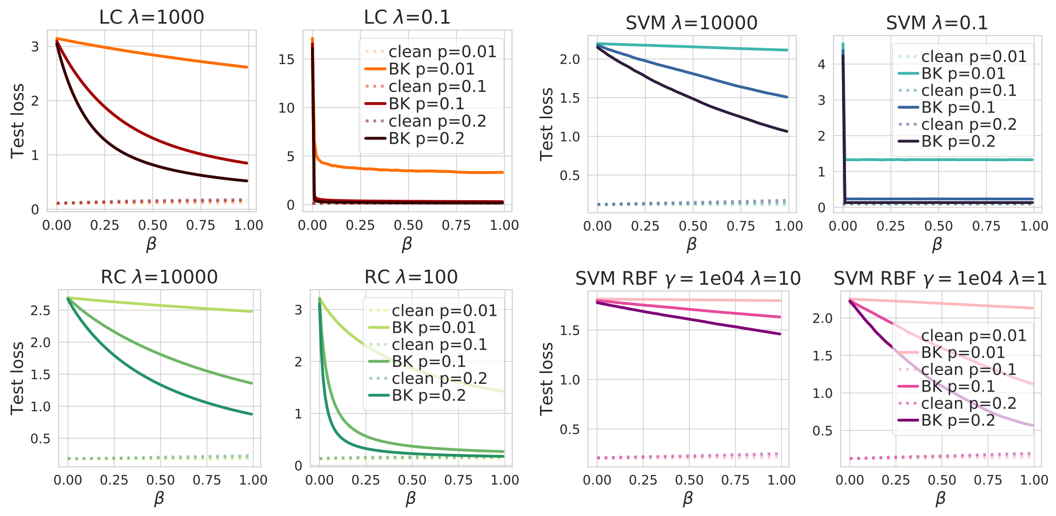


Figure 10: Backdoor learning curves for different classifiers trained on CIFAR10 *airplane vs frog*. See the caption of Figure 7 for further details.

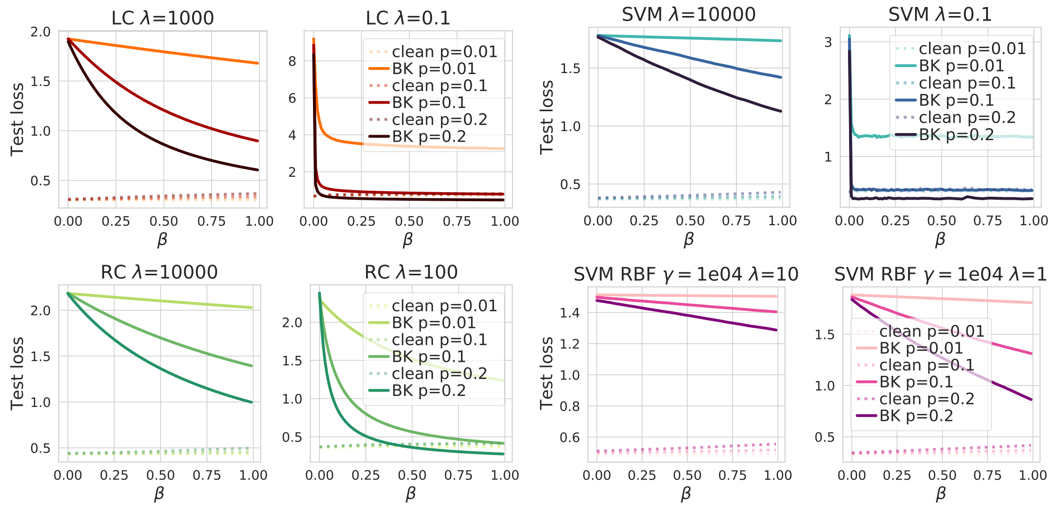


Figure 11: Backdoor learning curves for different classifiers trained on CIFAR10 *bird vs dog*. See the caption of Figure 7 for further details.

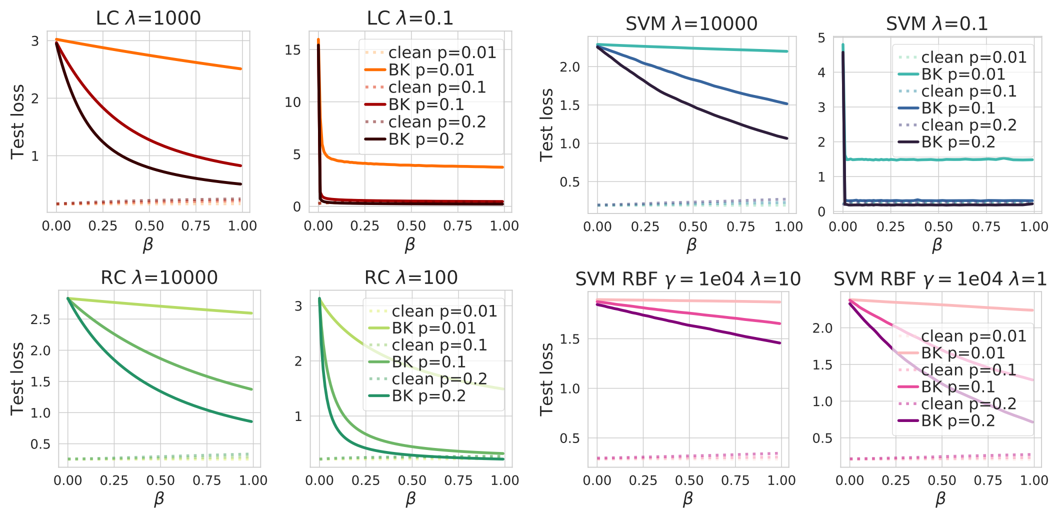


Figure 12: Backdoor learning curves for different classifiers trained on CIFAR10 *airplane vs truck*. See the caption of Figure 7 for further details.

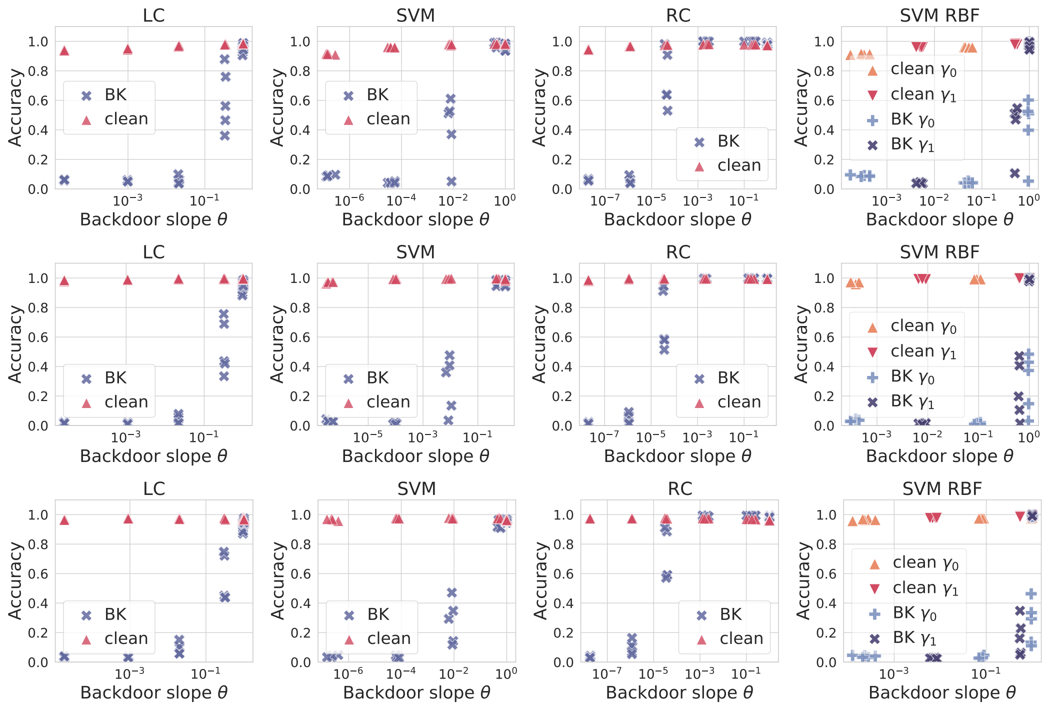


Figure 13: Backdoor slope θ vs backdoor (BK) effectiveness (blue) and clean accuracy (red) on MNIST 7 vs 1 (top), 3 vs 0 (middle) and 5 vs 2 (bottom). We have represented with a triangle (plus sign) the performance obtained with γ_0 on the clean (backdoored) dataset, and with a inverted triangle (cross), the ones obtained with γ_1.

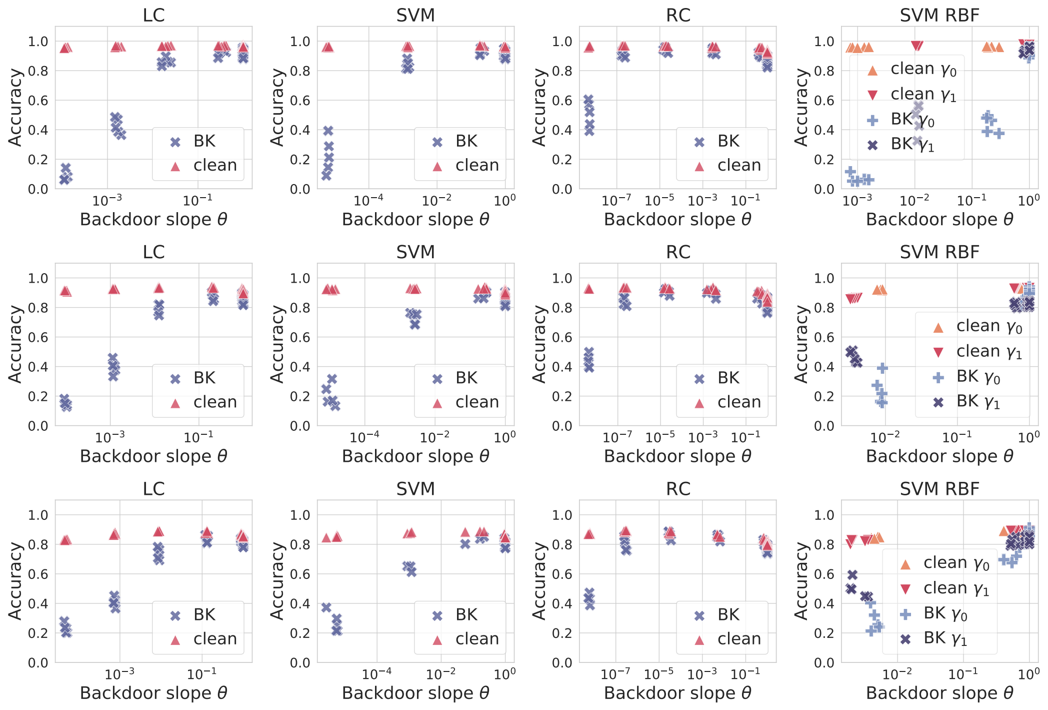


Figure 14: Backdoor slope vs backdoor (BK) effectiveness on CIFAR10 *airplane vs frog* (top), *airplane vs truck* (middle) and *bird vs dog* (bottom). See the caption of Figure 13 for further details.

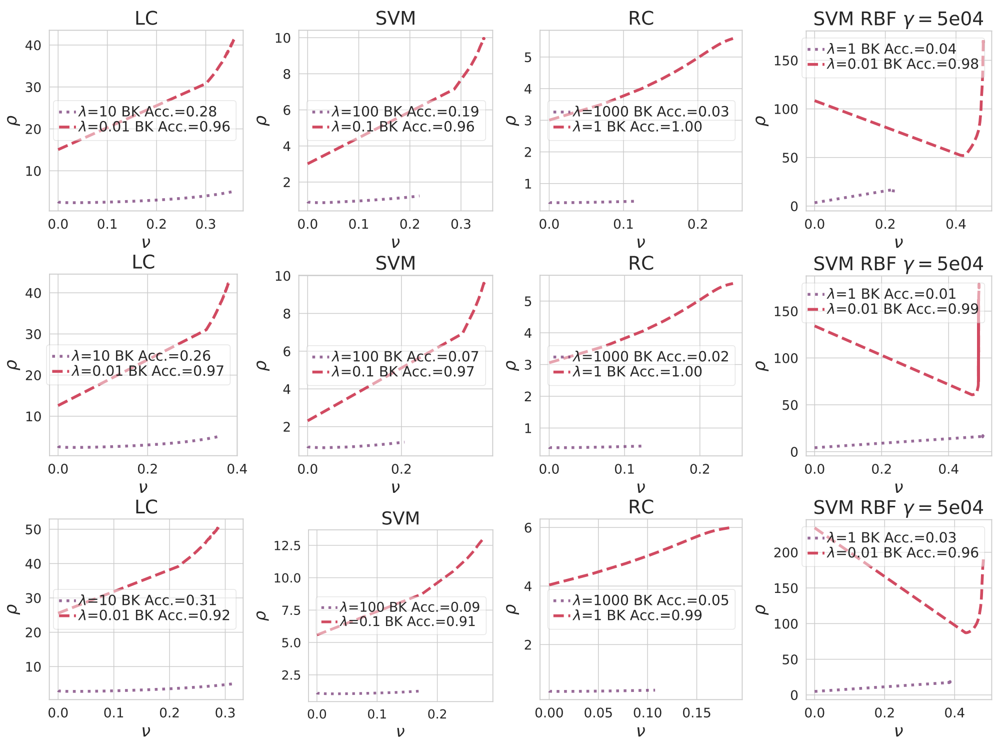


Figure 15: Backdoor weight deviation for different classifiers trained on MNIST 7 vs 1 (top), 3 vs 0 (middle) and 5 vs 2 (bottom). We specify regularization parameter λ and backdoor (BK) accuracy for each setting in the legend of each plot.

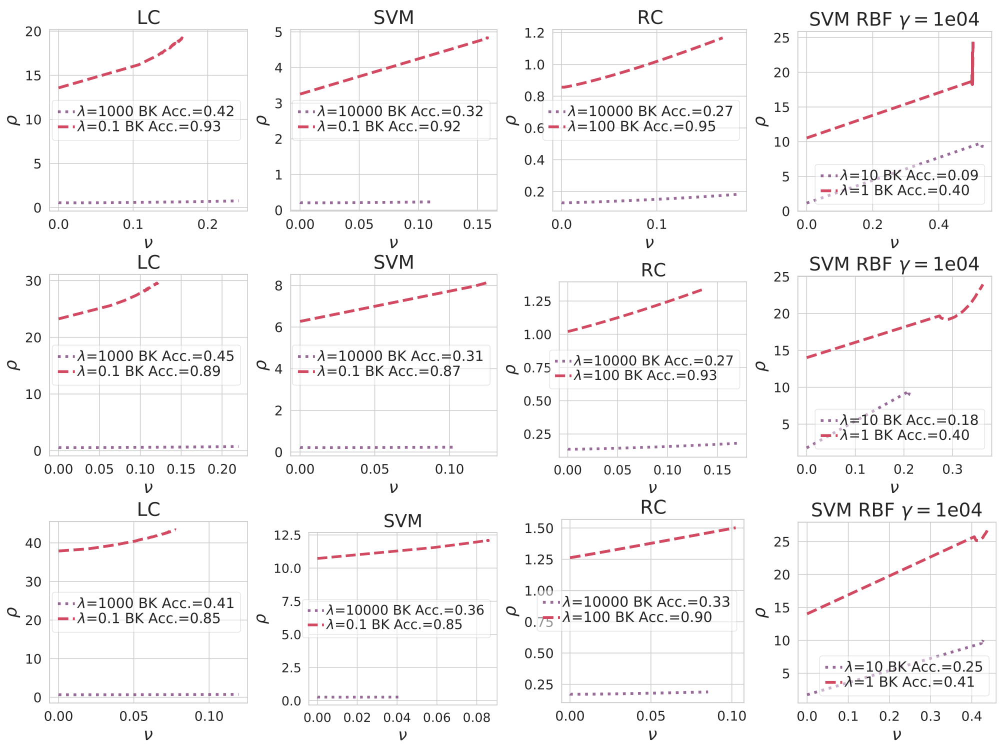


Figure 16: Backdoor weight deviation for different classifiers trained on CIFAR10 *airplane vs frog* (top), *airplane vs truck* (middle), and *bird vs dog* (bottom). We specify regularization parameter λ and backdoor (BK) accuracy for each setting in the legend of each plot.

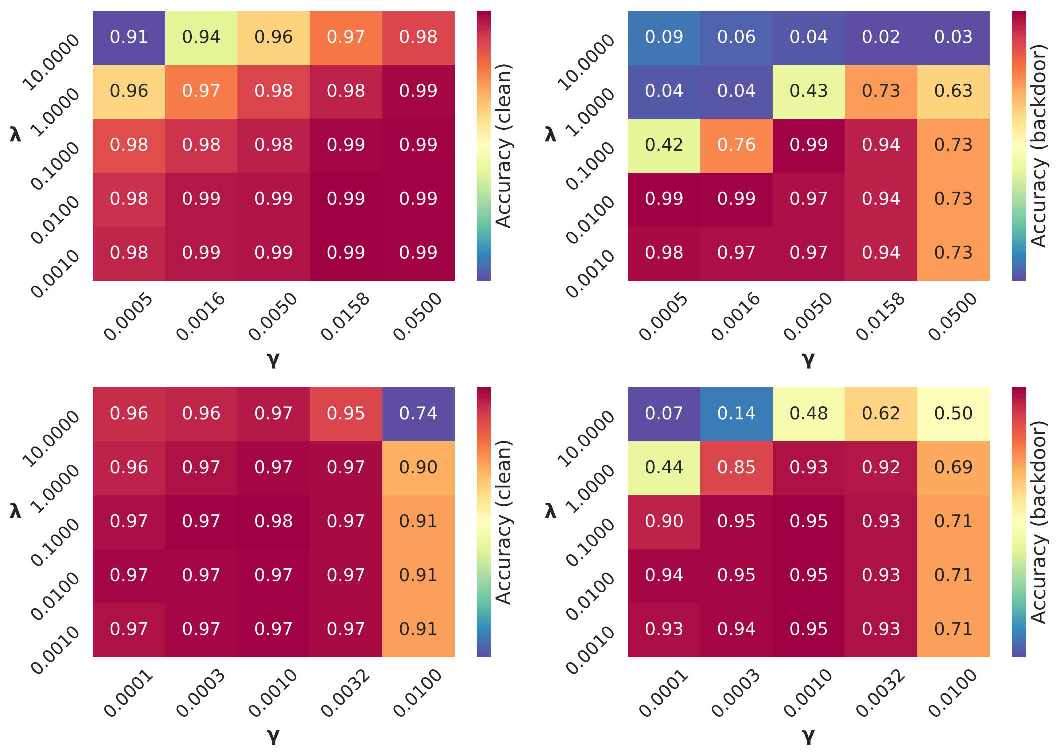


Figure 17: Influence of γ (x-axis) and λ (y-axis) on the backdoor effectiveness (left) and clean accuracy (right) for MNIST 7 vs 1 (top row) and CIFAR10 *airplane vs frog* (bottom row). The plots show that there are hyperparameter configurations for which clean accuracy is high (red regions on the left plots), while the accuracy on the backdoored points is low (blue regions on the right plots). Our results thus suggest backdoor robustness as an additional criterion for hyperparameter tuning.

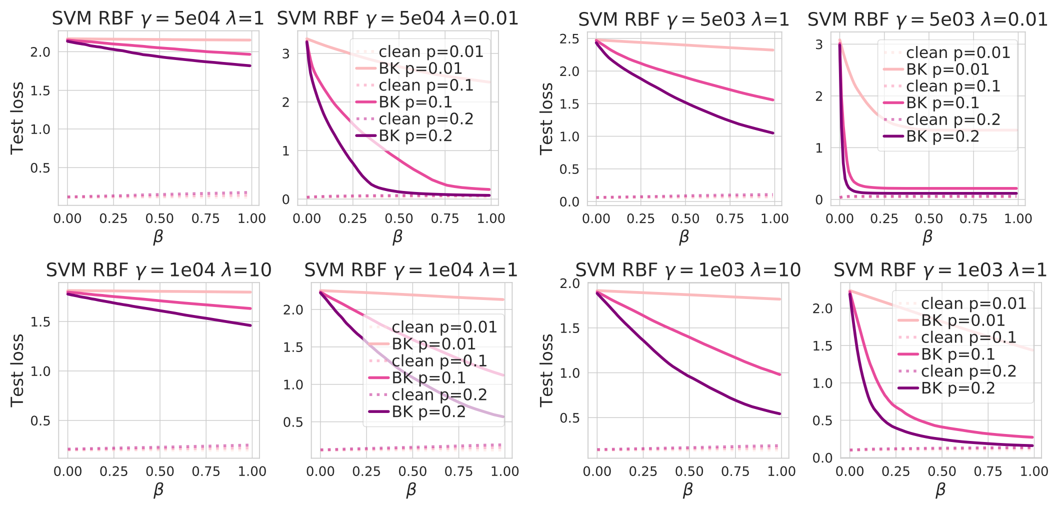
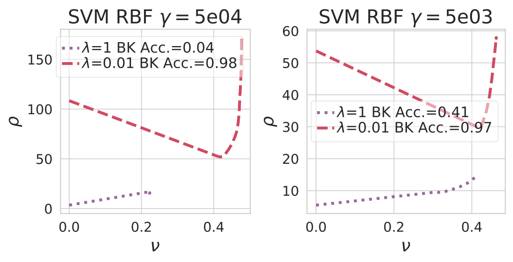
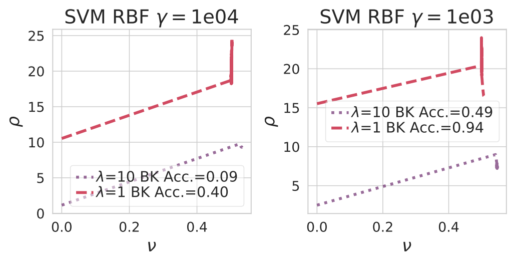


Figure 18: Backdoor learning curves for MNIST 7 vs 1 (top row) and CIFAR10 *airplane vs frog* (bottom row). The results for γ_0 (γ_1) are reported in the left (right) plot.



(a) MNIST 7 vs 1.



(b) CIFAR10 *airplane vs frog*.

Figure 19: Backdoor weight deviation for MNIST 7 vs 1 (19a) and CIFAR10 *airplane vs frog* (19b). The results for γ_0 (γ_1) are reported in the left (right) plot.