Backdoor Learning Curves: Explaining Backdoor Poisoning Beyond Influence Functions

Antonio Emanuele Cinà[†*], Kathrin Grosse[‡], Sebastiano Vascon[†], Ambra Demontis[‡], Battista Biggio[‡]
Fabio Roli[‡], and Marcello Pelillo[†]
[†]University of Venice Venice, Italy, [‡]University of Cagliari Cagliari, Italy
[*]`antonioemanuele.cina@unive.it`

Abstract

Backdoor attacks inject poisoned samples during training to force 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 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 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 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.

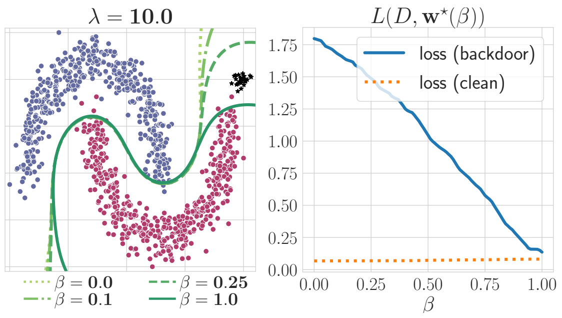
1. Introduction

Machine-learning models are vulnerable to backdoor poisoning [11, 18, 23], *i.e.*, the injection of poisoning training samples embedding a specific *trigger*, and assigned to an attacker-chosen class label. Under this attack, the classifier is misled to predict the attacker-chosen class whenever a sample containing the trigger is presented at test time. Although backdoor attacks have been demonstrated in a plethora of scenarios [11, 18, 27], the main factors behind their effectiveness are still poorly understood.

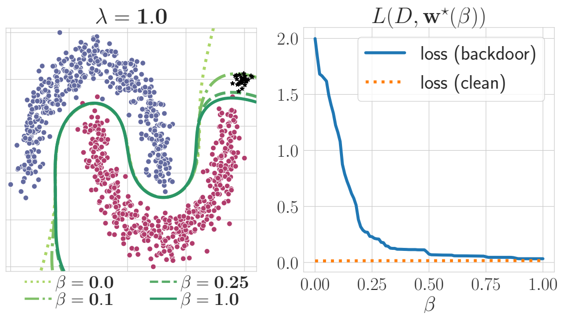
In this work, we propose a unifying framework to analyze the learning process and identify the main factors affecting the vulnerability of machine-learning models against backdoor attacks. To this end, we define the notion of *backdoor learning curves* (Sect. 2) by formulating backdoor learning as an incremental learning problem. More specifically, these curves show how the loss on backdoored points changes depending on the exposure to backdoors during training. To quantify how susceptible a model is to a backdoor attack, we introduce the notion of *backdoor learning slope*, a concept related to influence functions [14]. Intuitively, the higher the slope is, the quicker the model learns to incorporate the backdoor sample, thus being more vulnerable to such an attack. Our experimental analysis (Sect. 3) shows that the main factors that influence how fast the backdoor is learned are: (i) the fraction of backdoor samples injected into the training data, and (ii) the complexity of the target model, controlled via its hyperparameters. We surprisingly show 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. This behavior may be used to select more robust hyperparameter configurations against backdoor attack, thus inspiring novel defense mechanisms. We conclude the paper by discussing related work (Sect. 4), conclusions and future research developments (Sect. 5).

2. Backdoor Learning Curves

Before we introduce backdoor learning curves and the related concept of backdoor learning slope, we define notation. The input data is $\boldsymbol{x} \in \mathbb{R}^d$ and their labels are $y \in \{1, .., c\}$, where c is the number of classes. The untainted, clean training data are $\mathcal{D}_{\text{tr}} = (\boldsymbol{x}_i, y_i)_{i=1}^n$, and the backdoor samples injected into the training set are $\mathcal{P}_{\text{tr}} = (\hat{\boldsymbol{x}}_j, \hat{y}_j)_{j=1}^m$. 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 includes any regularization term that may be optimized during training. Accordingly, the loss attained by the classifier on the back-



(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) and 1 (right)). Each setting is visualized using two plots, where the left shows the data (dots) and decision surface for different β-values (green lines). The right plot shows the backdoor learning curve, where the y-axis depicts the test 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.

door samples presented at test time is given as

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

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

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

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 [4, 12]. 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 $\boldsymbol{w}^\star(\beta)$.

Backdoor Learning Curves. We can now define the *backdoor learning curve* as the curve showing the behavior of the loss $L(\mathcal{P}_{\rm ts}, \boldsymbol{w}^\star(\beta))$ 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 how fast an untainted classifier can be poisoned is thus expressible in terms of the gradient of the backdoor learning curve at $\beta = 0$. In mathematical terms, we write

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

As suggested from previous work in incremental learning [4], we assume that, while increasing β, the solution maintains the optimality (Karush-Kuhn-Tucker, KKT) conditions intact. This equilibrium implies that $\nabla_\beta \nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}^\star) + \frac{\partial \boldsymbol{w}^\star}{\partial \beta} \nabla^2_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}^\star) = \boldsymbol{0}$, and based on this condition, we were able to obtain the derivative of interest. This derivative, or Eq. 3, also corresponds to the sum of the pairwise influence function values $\mathcal{I}_{\rm up,loss}(\boldsymbol{x}_{\rm tr}, \boldsymbol{x}_{\rm ts})$ used by Koh and Liang [14].

However, directly using the gradient of the loss wrt. β has 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

$$\theta = -\frac{2}{\pi} \arctan\left(\left. \frac{\partial L}{\partial \beta} \right|_{\beta=0} \right) \in [-1, 1] \,, \tag{4}$$

where we use the negative sign correlate positive values 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 $[-1, 1]$.

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 when increasing the fraction of injected poisoning points, the classifier learns the backdoor faster. In particular, the backdoor is learned 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 increasing their complexity. Before we discuss these results in detail, however, we describe the experimental setup.

3.1. Experimental Setup

We first detail the datasets, then the models with their hyperparameters and finally the backdoor attacks and the applied triggers we implemented.

Datasets. In our analysis, we consider the CIFAR10 [15] dataset. Analogous to prior work [23], we chose *airplane vs frog*, *bird vs dog*, and *airplane vs truck*. In the following section, we focus on the results of one pair (*airplane vs frog* on CIFAR10). Appendix A contains the results of the other pairs and additional results on MNIST.

Models, Training and Hyperparameters. We consider various models to thoroughly analyze how learning a backdoor affects a model. More specifically, we analyze the results for Support Vector Machines (SVMs) with a Radial Basis Kernel (RBF). The results using linear SVM (SVM), Logistic Regression (LC), and Ridge Classifiers (RCs) in the Appendix. Analogously to the setting by Saha et al. [23], we train all models in a transfer learning setting with pre-trained Alexnet [16] used as a feature extractor. The hyperparameter choices are motivated in Appendix A.

Backdoor Attacks. We add a trigger to 10% of the training data if not stated otherwise. The trigger is a random 8×8 patch to strengthen the attack [23]. We replace the lower right corner of the image pixels with the trigger pattern, as done in [11]. Visual examples for samples with and without trigger can be found in the Appendix. After training, the presence of a trigger forces the backdoored model to predict the i-th class as class $(i+1)\%2$ [11]. However, our study

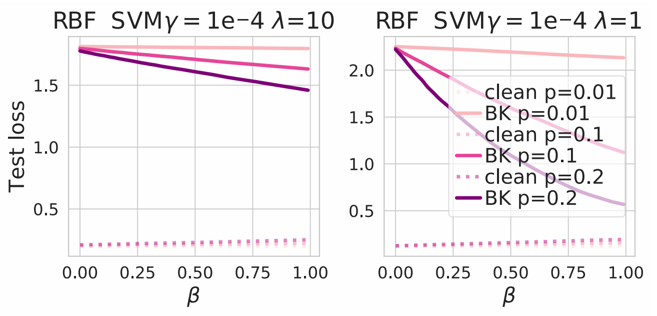


Figure 2: Backdoor learning curves for SVM on CIFAR10 *airplane vs frog* with $\gamma = 1e{-}04$. Left plot shows $\lambda = 10$, right plot $\lambda = 1$. Darker lines represent a higher fraction of poisoning samples p injected into the training set. We report the loss on backdoored (BK) test (solid line), and the loss for clean test samples (dashed line).

also encompasses linear models that are unable to associate the same trigger pattern to two different classes. Thus, in contrast to previous approaches [11], we use a separate trigger for each base-class i.

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 for RBF SVM trained on CIFAR10, 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. The accuracy for benign samples, however, does not change as p grows larger. We conclude that less regularized, or 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 [14] is the partial derivative of the backdoor learning curve at the point $\beta = 0$. We investigate the backdoor effectiveness through the lens of

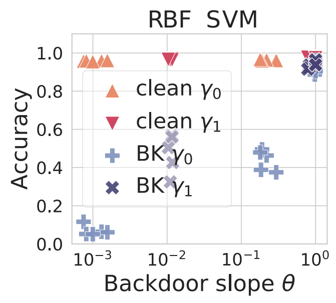


Figure 3: Backdoor slope θ vs backdoor (BK) effectiveness and clean accuracy on CIFAR10 *airplane vs frog*. We set $\gamma_0 = 1\mathrm{e}{-04}$ (orange triangle for clean data, light blue plus for data with trigger) and $\gamma_1 = 1\mathrm{e}{-03}$ (red inverted triangle for clean data, dark blue x for data with trigger).

the influence functions when the fraction of poisoning at $p = 0.1\%$, as studied by Gu et al. [11]. 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 backdoor effectiveness as the percentage of samples with trigger that mislead the classifier. We report results for $\gamma \in \{1\mathrm{e}{-04}, 1\mathrm{e}{-03}\}$. In Figure 3, the blue '+' and 'x' lie roughly on the bisect line. This means that backdoor accuracy tends to increase when the backdoor slope increases. Thus a higher slope implies higher vulnerability to backdoor attacks. As the clean accuracy (red and orange triangles) in Figure 3 is rather high regardless of the backdoor slope, there is a region where backdoor effectiveness is low and benign accuracy is high (left side of the plot). These findings, along with the extended analyses reported in Appendix A, show that the best trade-off is achieved when λ and γ are small, *i.e.*, when the classifier is strongly regularized, which makes it harder to learn the backdoor pattern.

A deeper understanding of backdoor learning. We have shown that more complex, weakly-regularized models are more vulnerable to backdoor attacks, as they are able to fit the backdoor samples more easily. Further experiments confirming this hypothesis are reported in Appendix A, where we consider more datasets, classifiers and hyperparameters. Moreover, in Appendix B we show that less complex models tend to monotonically increase their complexity during the backdoor learning process to preserve high accuracy for both clean and backdoor samples. These results are also supported in our explainability analysis reported in Appendix C.

Summarizing, our findings suggest that an appropriate choice of complexity-regulating hyperparameters such as λ can help in building new defences or improving existing approaches in terms of backdoor robustness.

4. Related Work

We first review the literature about backdoors and then the related work in the area of influence functions.

Backdoors Although introduced recently [11, 5], there is a plethora of backdoor attacks and backdoor defenses. An overview can for example be found in the survey by Gao et al. [9] or Goldblum et al. [10]. Few works study backdoors not as attack or defense but rather as a phenomenon. For example, Frederickson et al. [8] 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. [25] study the vulnerability of ensembles versus individual classifiers. However, we focus on the robustness or vulnerability of individual classifiers. Carnerero-Cano et al. [3] study the effect regularization has on poisoning, however focusing on indiscriminate attacks that reduce accuracy (e.g., not on backdoors). Finally, Baluta et al. [1] 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. We are the first to experimentally show and outline a relationship between model complexity and backdoor effectiveness.

Influence functions stem from robust statistics [7] and were later used as a tool to measure the influence of training points on the classification output on for example an SVM [6] or deep learning [14]. However, Basu et al. [2] showed later that influence functions are more reliable in shallow models as they are fragile in deep learning.

5. Conclusions

In this work, we have presented a framework to analyze the factors that influence the effectiveness of backdoor poisoning. Our work shows that this backdoors effectiveness inherently depends (i) on the number of backdoor samples injected into the training dataset and (ii) the target model's complexity. Furthermore, we show a region in the hyperparameter space in which the accuracy on clean test samples is still high while the accuracy on triggered backdoors is low. Our experiments were carried out on convex learners. It remains thus an open question whether our findings generalize to deeper models. 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. We also leave for future work the integration of factors like the amount of clean and poisoned data, the size and shape of the trigger, or the size of the input space.

Acknowledgments

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] Teodora Baluta, Shiqi Shen, Shweta Shinde, Kuldeep S Meel, and Prateek Saxena. Quantitative verification of neural networks and its security applications. In *CCS*, 2019. 4

[2] Samyadeep Basu, Philip Pope, and Soheil Feizi. Influence functions in deep learning are fragile. In *ICLR*, 2021. 4

[3] 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. 4

[4] Gert Cauwenberghs and Tomaso Poggio. Incremental and decremental support vector machine learning. *Advances in neural information processing systems*, pages 409–415, 2001. 2

[5] 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. 4

[6] 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. 4

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

[8] 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. 4

[9] 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. 4

[10] 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. 4

[11] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Evaluating backdooring attacks on deep neural networks. *IEEE Access*, 7:47230–47244, 2019. 1, 3, 4

[12] 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. 2

[13] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. 6

[14] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *ICML*, 2017. 1, 2, 3, 4

[15] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 3

[16] 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. 3

[17] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010. 6

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

[19] 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. 6

[20] The pandas development team. pandas-dev/pandas: Pandas, Feb. 2020. 6

[21] 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. 6

[22] 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. 6

[23] Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. In *AAAI*, 2020. 1, 3

[24] 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. 6

[25] 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. 4

[26] Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021. 6

[27] Zhaohan Xi, Ren Pang, Shouling Ji, and Ting Wang. Graph backdoor. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*, 2021. 1

Appendix

A. Additional Experimental Results

In this section, we present additional experimental results that confirm our claims.

Datasets. In our analysis we also consider the MNIST [17] dataset. We chose 7 vs 1, 3 vs 0, and 5 vs 2, as on these pairs our models exhibited the highest clean accuracy.

Hyperparameters. We vary λ in $\{1e{-}03, 1e{-}02, 1e{-}01, 1, 1e{+}01, 1e{+}02, 1e{+}03, 1e{+}04\}$. Concerning the RBF kernel, we let γ take uniformly spaced values on a log scale in $[5e{-}04, 5e{-}02]$ for MNIST and $[1e{-}04, 1e{-}02]$ for CIFAR. Furthermore, we chose $\lambda \in \{1e{-}03, 1e{-}02, 1e{-}01, 1, 1e{+}01\}$ 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 $\gamma = 5e{-}02$ and $\lambda = 1e{+}01$, where the accuracy on samples without trigger is only 75%.

Implementation. We rely on public libraries to implement our models [22, 21], standard procedures like reading and accessing data [20], mathematical computations like for example norms or gradients [24, 19] and to create the plots in our paper [13, 26].

Backdoor Learning Curves. In the paper, we have shown the backdoor learning curves only for the SVM RBF on CIFAR10, here, we report them for all the classifiers and datasets considered in our 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 $\gamma = 5e{-}03$ (MNIST) or $\lambda \in \{10, 1\}$ and $\gamma = 1e{-}03$ (CIFAR10).

Moreover, we compare the results obtained on the class pairs considered in the paper (*airplane vs frog* on CIFAR10), shown in Figure 7, with the one obtained on different pairs.

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 MNIST dataset for class pairs: 7 vs 1 (Figure 4), 3 vs 0 (Figure 5) and 5 vs 2 (Figure 6). We further show the results obtained on the class pairs *bird vs dog* (*airplane vs truck*) of the CIFAR10 dataset in Figure 8 (Figure 9).

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

Backdoor Slope. In Figure 10 and 11, 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.

B. Backdoor Impact on Learned Parameters

In this section, we investigate the influence of learning backdoors on classifier complexity.

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,

$$\rho = \|\boldsymbol{w}_\beta\| \in [0, \infty), \text{ and } \nu = \frac{1}{2}\left(1 - \frac{\boldsymbol{w}_0^\top \boldsymbol{w}_\beta}{\|\boldsymbol{w}_0\|\|\boldsymbol{w}_\beta\|}\right) \in [0, 1]\,. \tag{5}$$

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 B.1, 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.

B.1. Experiments

We investigate the two measures proposed in Section B, 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 12 for MNIST and in Figure 13 for CIFAR10. 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.

C. 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 14.

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, 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 14 confirm our results from Section B.1 regarding the increase in complexity due to the backdoor.

Figure 15 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.

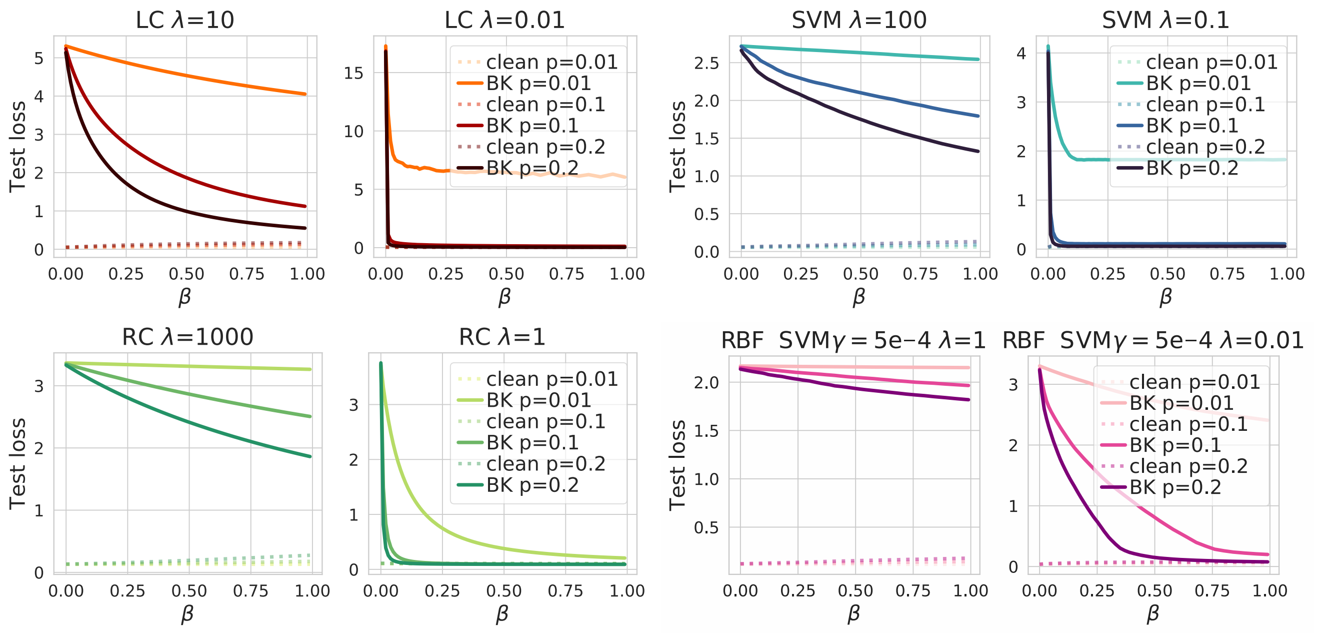


Figure 4: Backdoor learning curves for different classifiers trained on MNIST 7-1. Darker lines represent a higher fraction of poisoning samples p injected into the training set. We report with solid (dashed) lines the loss on the backdoored (clean) test dataset.

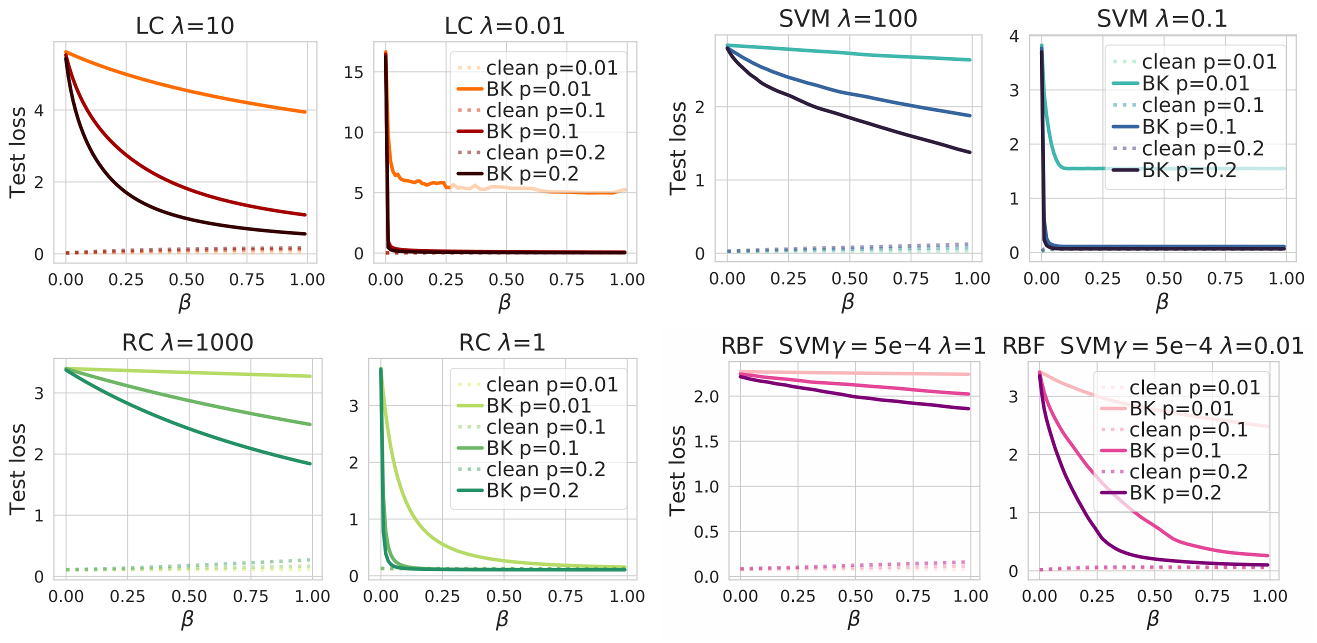


Figure 5: Backdoor learning curves for different classifiers trained on MNIST 3-0. See the caption of Figure 4 for further details.

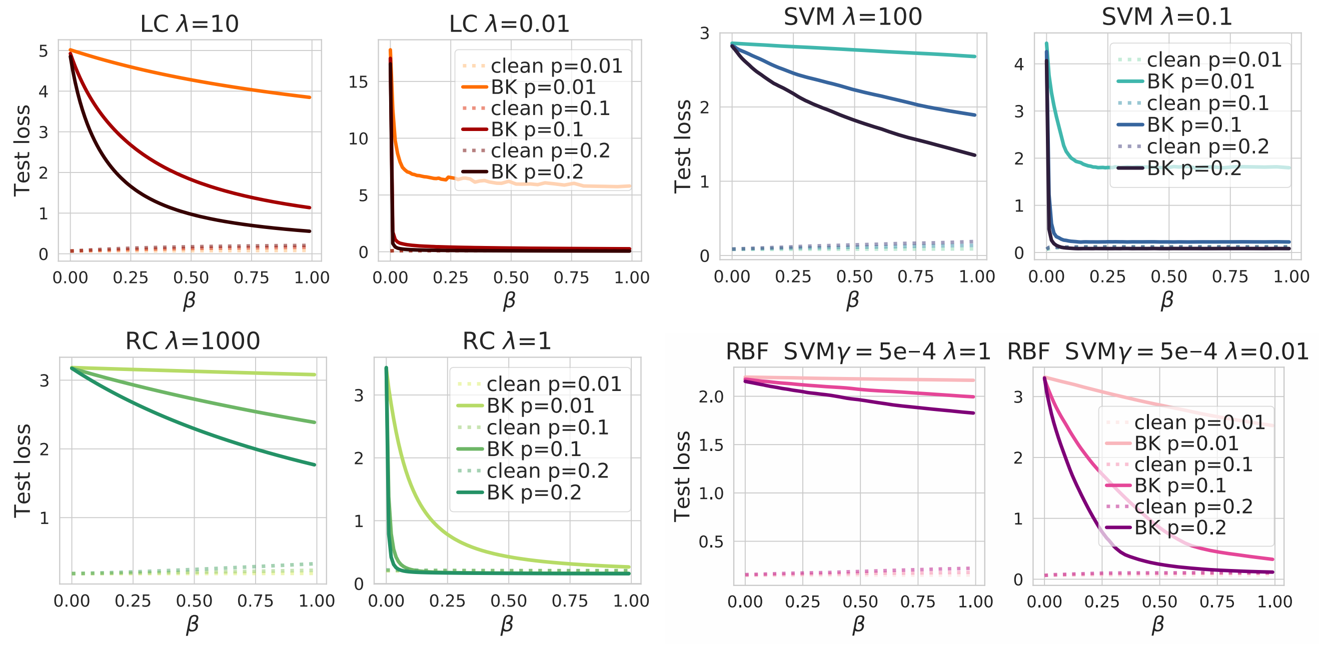


Figure 6: Backdoor learning curves for different classifiers trained on MNIST 5-2. See the caption of Figure 4 for further details.

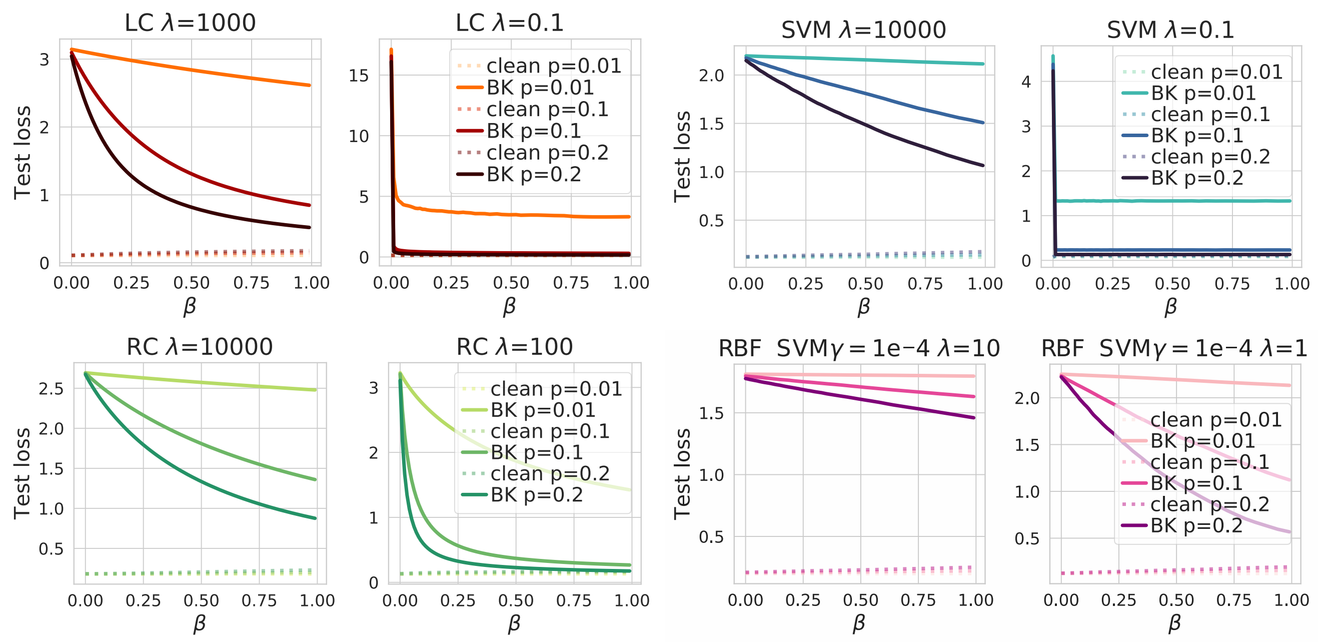


Figure 7: Backdoor learning curves for different classifiers trained on CIFAR10 *airplane vs frog*. See the caption of Figure 4 for further details.

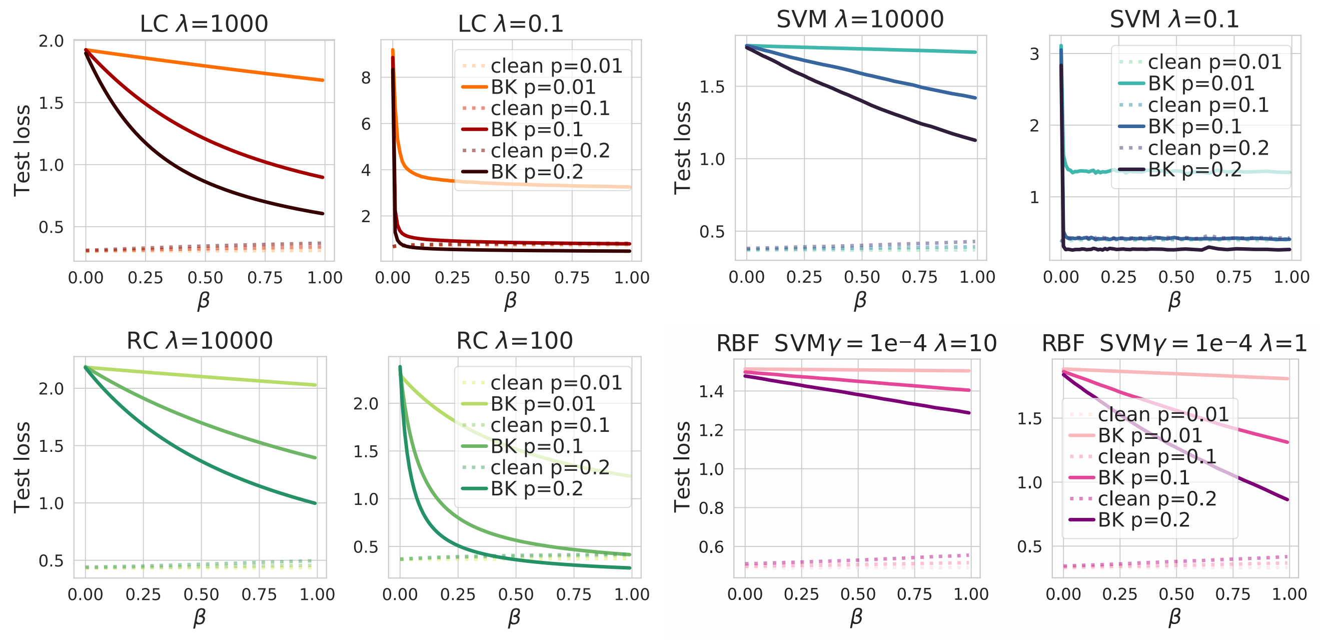


Figure 8: Backdoor learning curves for different classifiers trained on CIFAR10 *bird vs dog*. See the caption of Figure 4 for further details.

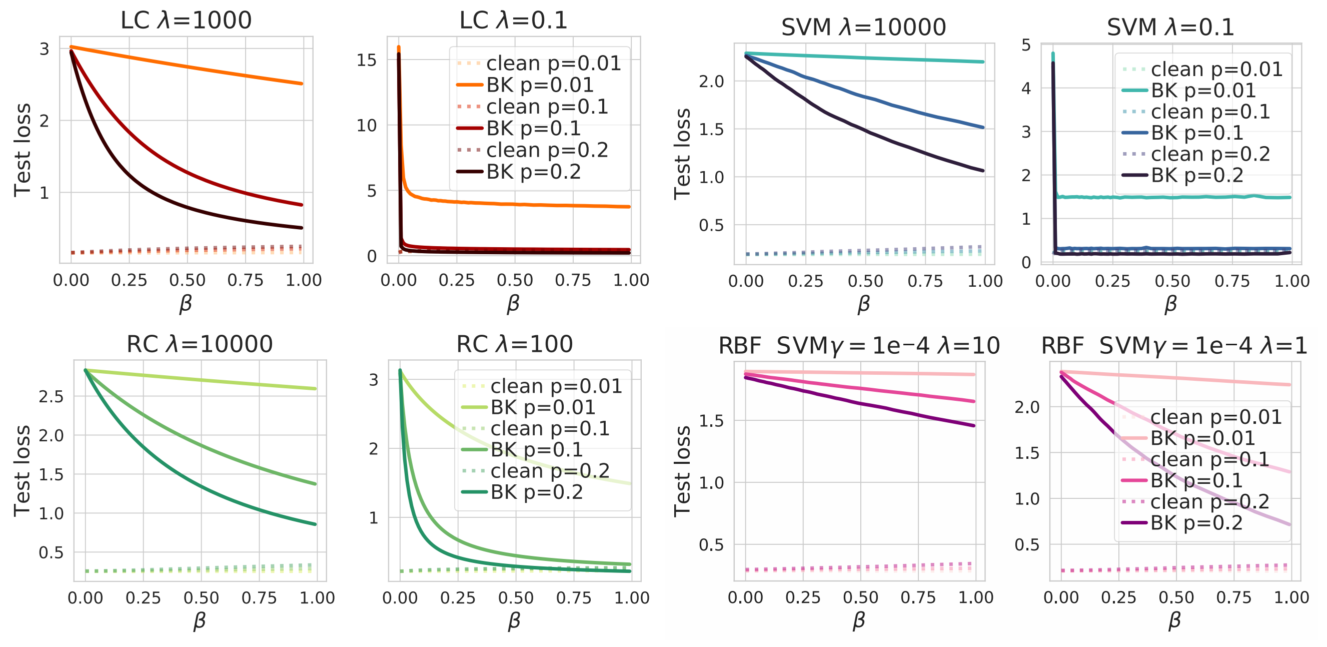


Figure 9: Backdoor learning curves for different classifiers trained on CIFAR10 *airplane vs truck*. See the caption of Figure 4 for further details.

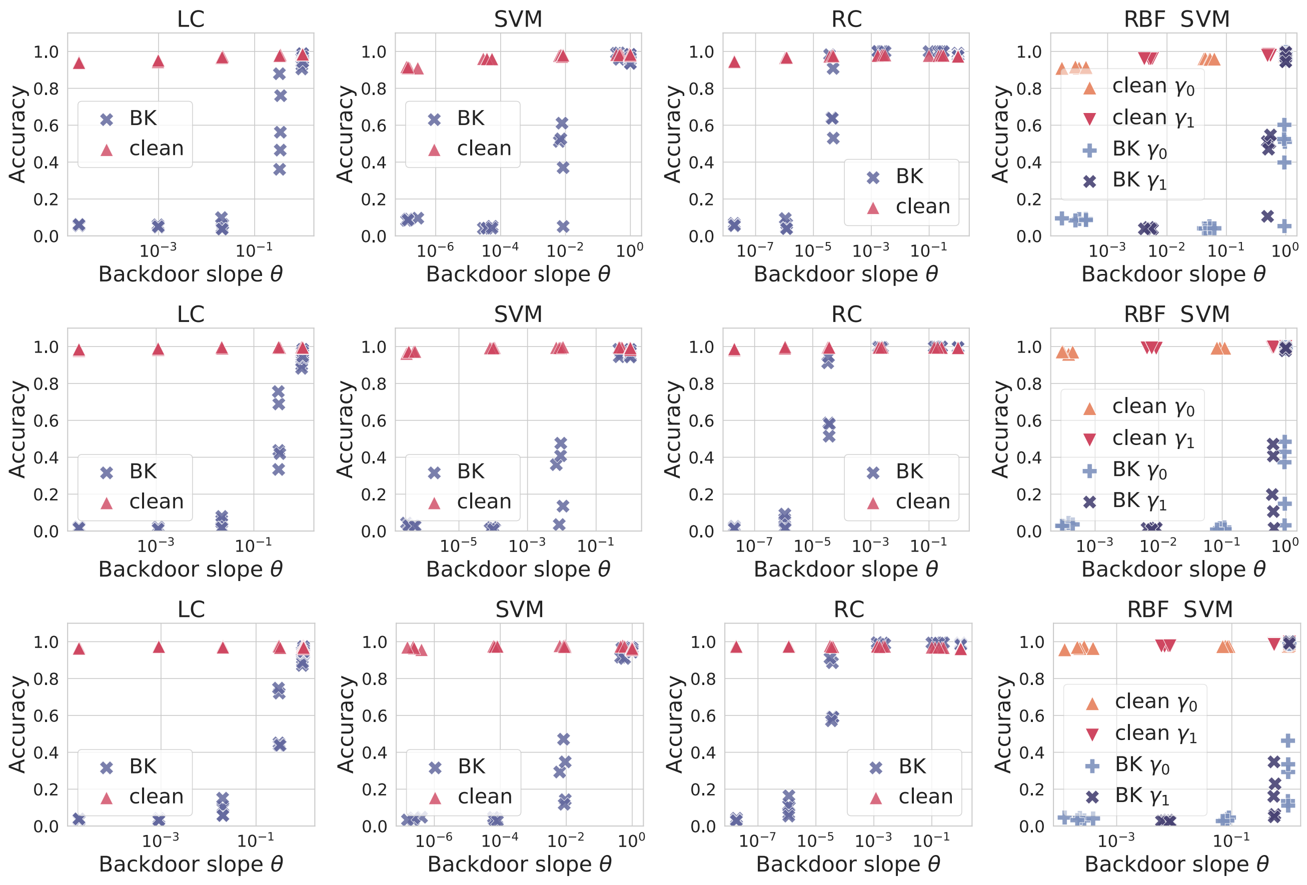


Figure 10: Backdoor slope θ vs backdoor (BK) effectiveness (blue) and clean accuracy (red) on MNIST 7 vs 1 (top), 3 vs 0 (middle) and 5 vs 2 (bottom). We have represented with a triangle (plus sign) the performance obtained with γ_0 on the clean (backdoored) dataset, and with a inverted triangle (cross), the ones obtained with γ_1.

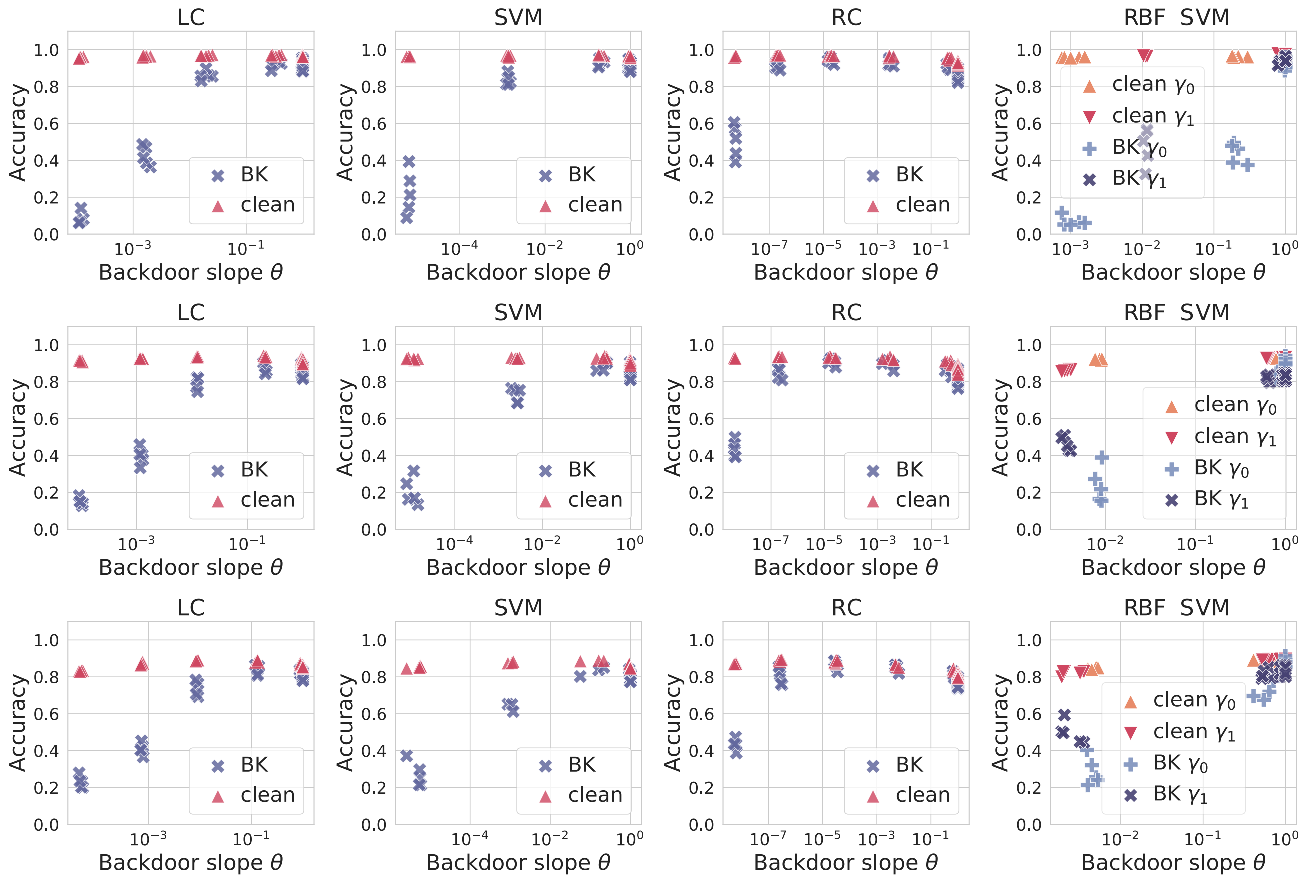


Figure 11: Backdoor slope vs backdoor (BK) effectiveness on CIFAR10 *airplane vs frog* (top), *airplane vs truck* (middle) and *bird vs dog* (bottom). See the caption of Figure 10 for further details.

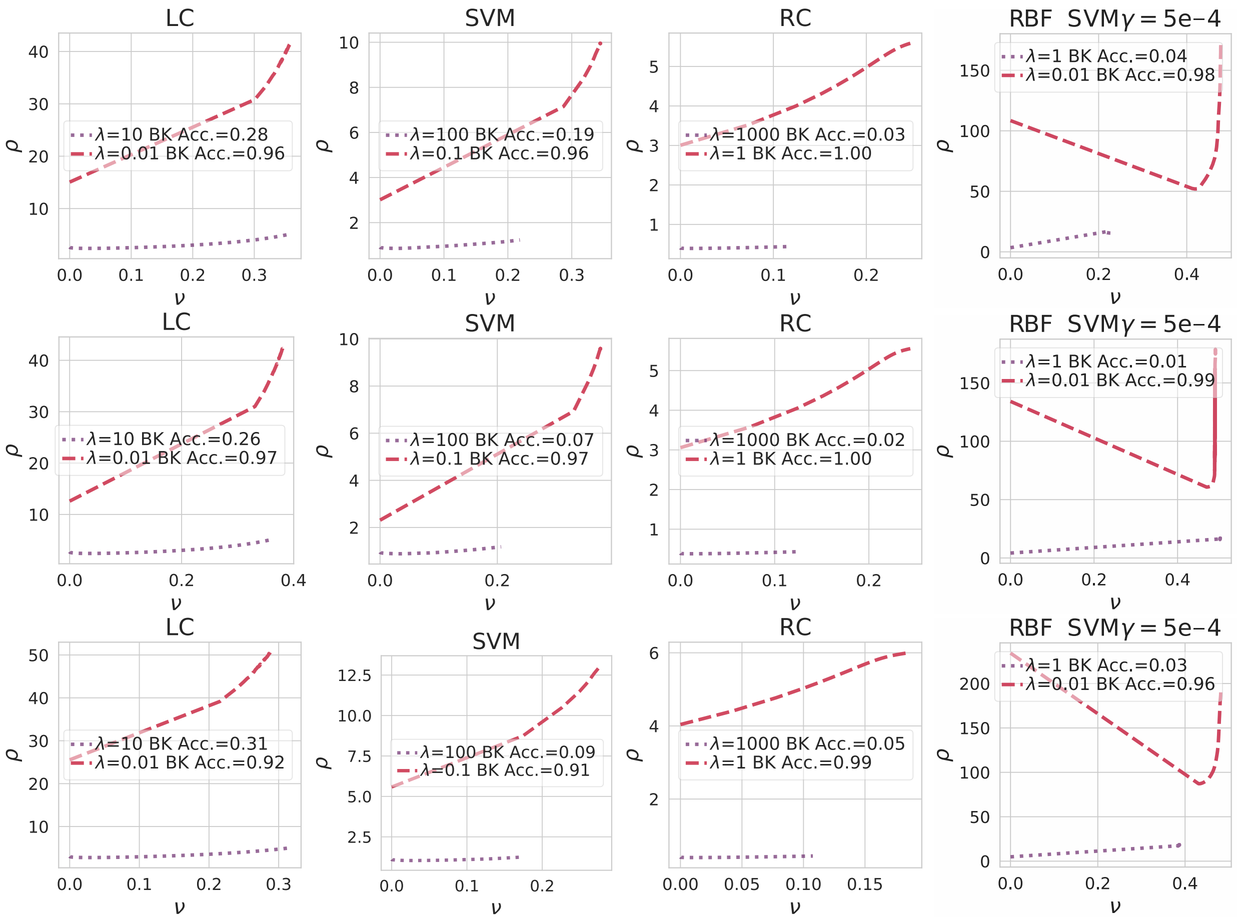


Figure 12: Backdoor weight deviation for different classifiers trained on MNIST 7 vs 1 (top), 3 vs 0 (middle) and 5 vs 2 (bottom). We specify regularization parameter λ and backdoor (BK) accuracy for each setting in the legend of each plot.

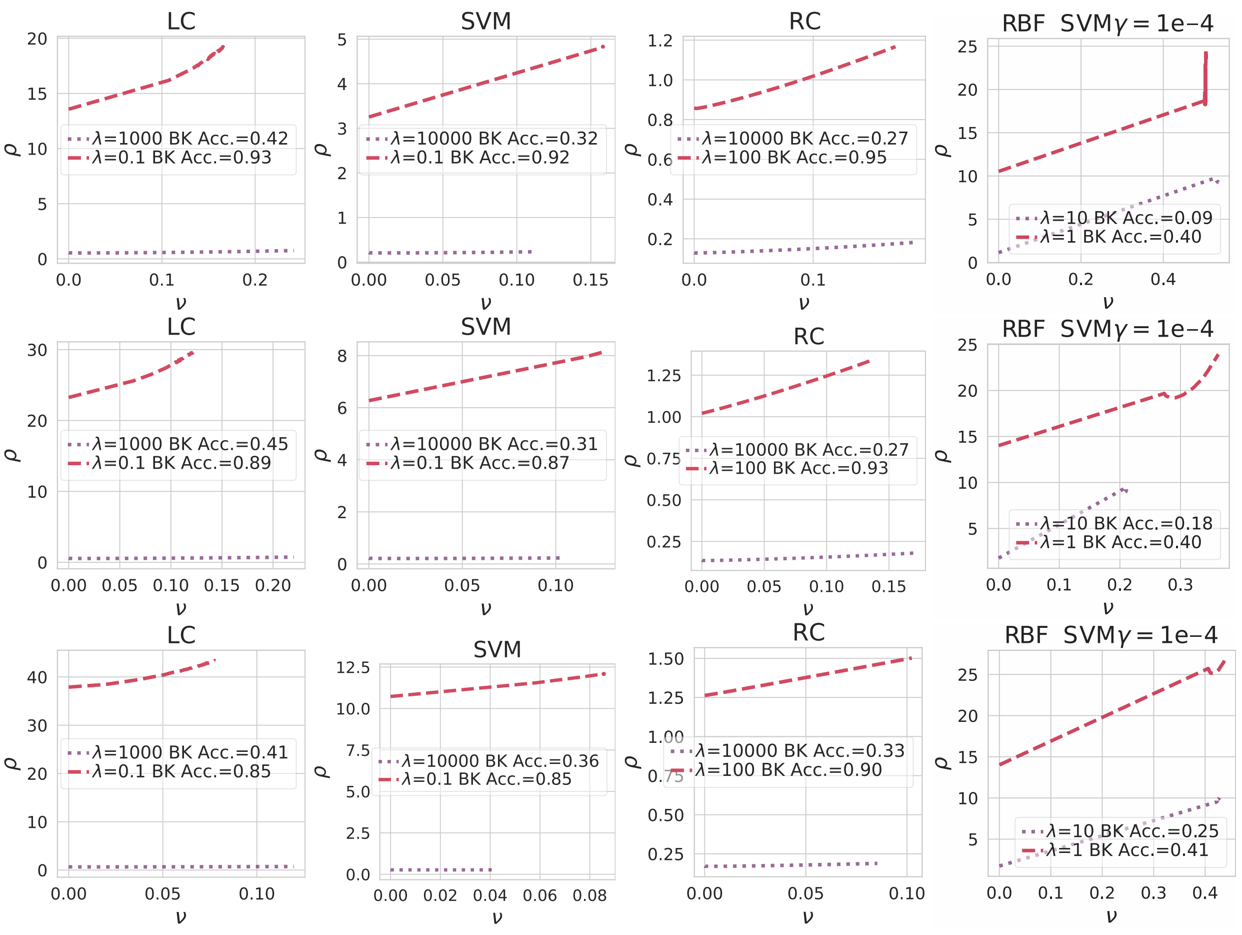


Figure 13: Backdoor weight deviation for different classifiers trained on CIFAR10 *airplane vs frog* (top), *airplane vs truck* (middle), and *bird vs dog* (bottom). We specify regularization parameter λ and backdoor (BK) accuracy for each setting in the legend of each plot.

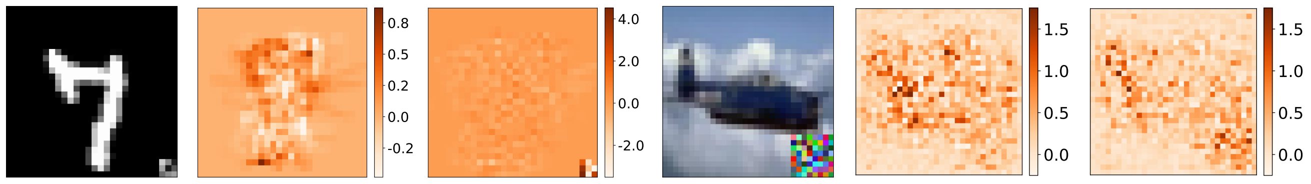


Figure 14: The first half of the plots consider the case of MNIST 7-1, and the second half is for CIFAR10 *airplane vs frog*. For each block we have: the poisoned test sample under consideration (left); the gradient of the untainted SVM decision's boundary with respect to the input (middle); the gradient 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.

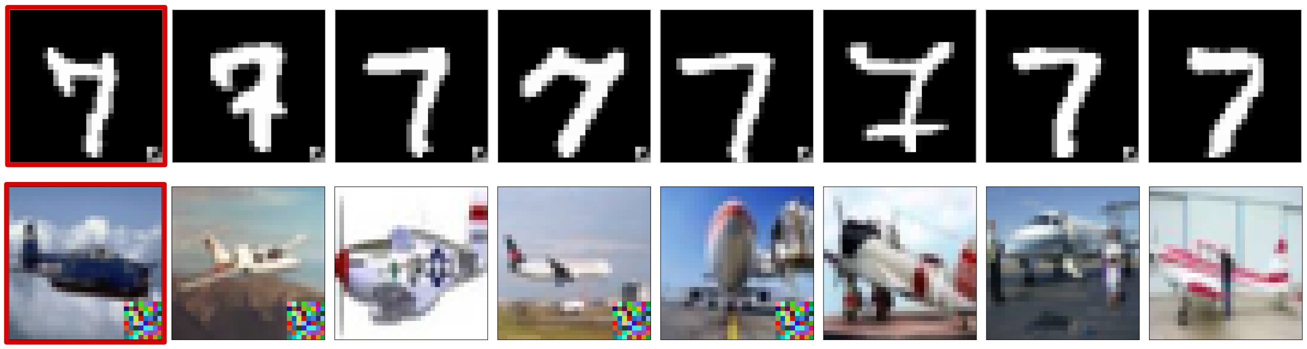


Figure 15: We report the top 7 most influent samples from the poisoned SVM on the prediction of the samples with the red border.