Tuning for Two Adversaries: Enhancing the Robustness Against Transfer and Query-Based Attacks using Hyperparameter Tuning

Pascal Zimmer, Ghassan Karame

Ruhr University Bochum, Germany
pascal.zimmer@rub.de, ghassan@karame.org

Abstract

In this paper, we present the first detailed analysis of how training hyperparameters—such as learning rate, weight decay, momentum, and batch size—influence robustness against both transfer-based and query-based attacks.

Supported by theory and experiments, our study spans a variety of practical deployment settings, including centralized training, ensemble learning, and distributed training. We uncover a striking dichotomy: for transfer-based attacks, decreasing the learning rate significantly enhances robustness by up to 64%. In contrast, for query-based attacks, increasing the learning rate consistently leads to improved robustness by up to 28% across various settings and data distributions. Leveraging these findings, we explore—for the first time—the training hyperparameter space to jointly enhance robustness against both transfer-based and query-based attacks. Our results reveal that distributed models benefit the most from hyperparameter tuning, achieving a remarkable tradeoff by simultaneously mitigating both attack types more effectively than other training setups.

Code — https://github.com/RUB-InfSec/tuning_for_two_adversaries

Extended version — https://arxiv.org/abs/2511.13654

1 Introduction

Despite their growing popularity, machine learning systems remain vulnerable to relatively simple manipulations of their inputs (Szegedy et al. 2014; Biggio et al. 2013). In particular, adversarial examples pose a significant threat to the application of deep neural networks (DNNs) in safety-critical domains, including autonomous driving and facial recognition. Recent strategies, known as black-box attacks (Chen, Jordan, and Wainwright 2020; Andriushchenko et al. 2020; Chen et al. 2024), adopt a practical and realistic threat model in which the attacker lacks access to the classifier's internal details and training data and can either (1) build a surrogate model to generate adversarial examples that transfer to the target (called transfer-based attacks) in one-shot attacks, or (2) interact with the model through oracle access—by observing its outputs in response to specific inputs (aka. query-based attacks) as is the case in machine-learning-as-a-service (MLaaS) settings.

A range of defense strategies has been proposed to counter adversarial examples, with adversarial training (Goodfellow, Shlens, and Szegedy 2015) still regarded as the gold standard among training-time defenses. However, this method is computationally expensive, as it requires retraining the target model using adversarially perturbed *pre-generated* inputs. To reduce overhead, lightweight test-time defenses—such as input transformations (Guo et al. 2018)—have been explored. However, these methods frequently compromise performance on clean inputs when aiming for strong robustness. Moreover, the rapid evolution of machine learning research has led to defenses being developed in isolation for either transfer-based or query-based attacks. This raises significant concerns about their ability to effectively enhance robustness across the full spectrum of black-box attack scenarios.

Given the lack of bullet-proof solutions to thwart adversarial examples, a natural question is whether the training hyperparameters themselves present an opportunity in this setting, as they are inherently tied to the target model and are independent of specific attacks. While prior studies have shown that factors such as the scheduler, optimizer, and architecture have only minimal impact on robustness (Andreina, Zimmer, and Karame 2025), hyperparameters—such as learning rate, momentum, weight decay, and batch size—significantly influence model smoothness due to their regularizing effects, a property closely associated with robustness to adversarial attacks (Moosavi-Dezfooli et al. 2019; Zhang et al. 2024; Demontis et al. 2019). Unfortunately, to the best of our knowledge, no prior work has conducted a precise analysis of how hyperparameter tuning affects robustness against specific attacks in the black-box setting.

In this paper, we address this gap and present the first precise analysis of how training hyperparameters—such as learning rate, weight decay, momentum, and batch size—influence robustness against strong transfer-based and query-based black-box attacks. Our analysis spans a range of popular machine learning deployment scenarios, including centralized setups (with a single training and inference instance), ensemble learning, and distributed learning, where both training and inference are distributed across multiple nodes. We also account for different data distribution set-

tings, encompassing both independent and identically distributed (i.i.d.) and non-i.i.d. training data. More specifically, we aim to answer the following research questions:

RQ 1 To what extent do training hyperparameters influence robustness against transfer-based attacks?

RQ 2 Similarly, how do training hyperparameters influence robustness against query-based attacks?

RQ 3 Is there an instantiation that naturally lends itself to effective tuning against both transfer-based and query-based attacks?

Supported by theory and extensive experiments on CIFAR-10 and ImageNet, our results reveal a striking contrast. On the one hand, we find that decreasing the learning rate can significantly enhance robustness against transfer-based attacks—by up to 64%—with this improvement consistently holding across different ML deployment scenarios, including centralized, ensemble, and distributed setups. Surprisingly, we observe the opposite trend for query-based attacks: increasing the learning rate leads to a robustness gain of up to 28% across all considered deployments and data distributions. Last but not least, we explore how to effectively balance hyperparameter tuning to navigate the trade-offs between robustness to transfer-based and query-based attacks. Through an extensive search using the NSGA-II algorithm, we demonstrate that there exist well-chosen hyperparameter configurations that strike a strong trade-off, enhancing robustness against both transfer-based and query-based attacks, *when compared to state-of-the-art defenses*, such as adversarial training (Rebuffi et al. 2021; Gowal et al. 2021) and JPEG compression (Guo et al. 2018).

2 Background & Related Work

Black-Box Attacks

Transfer-based Attacks. In transfer attacks, adversaries train surrogate models and apply white-box attacks on them. Success depends on the similarity between the surrogate and target models. To improve transferability in black-box settings, various techniques have emerged, including momentum (Dong et al. 2018), input diversification (Wu et al. 2021), model ensembling, and sharpness-aware minimization (Chen et al. 2024). Notably, (Chen et al. 2024) showed that reducing sharpness and aligning gradients boosts transferability. Other studies explore how model architecture, capacity (Demontis et al. 2019), and regularization-based control of smoothness and gradients (Zhang et al. 2024) affect transferability bounds.

Query-based Attacks. Query-based attacks interact directly with the target model. Here, attackers are limited by a query budget Q and the granularity of accessible information. Score-based attacks (Chen et al. 2017; Tu et al. 2019; Andriushchenko et al. 2020; Ilyas et al. 2018) use model output scores or probabilities, while decision-based attacks (Chen, Jordan, and Wainwright 2020) rely only on the top-1 predicted label. Most attacks begin with locating the decision boundary via binary search, then use gradient estimation (Chen et al. 2017) or geometric strategies to approach the boundary while maintaining adversarial properties. SquareAttack (Andriushchenko et al. 2020), a state-of-the-art score-based method, employs square-shaped perturbations and a simple random search scheme.

Hyperparameter Tuning

Despite the theoretical insights, hyperparameter tuning in practice remains largely ad hoc with limited insights on how hyperparameter configurations influence black-box robustness (Bagdasarian and Shmatikov 2024).

Implicit Regularization. Training hyperparameters implicitly regularize neural networks by influencing the geometry of the learned solutions, often characterized by their flatness or sharpness—quantified via the largest eigenvalue of the Hessian (Kaur, Cohen, and Lipton 2022). In full-batch gradient descent, the sharpness of the converged solution scales inversely with the learning rate η (Cohen et al. 2021). This inverse relationship approximately holds for small-batch stochastic gradient descent (SGD) as well (Kaur, Cohen, and Lipton 2022). Other hyperparameters such as batch size B and weight decay λ also affect sharpness: increasing the ratio η/B tends to decrease the largest Hessian eigenvalue (Jastrzębski et al. 2018), while the product $\eta\lambda$ has been linked to an implicit form of Jacobian regularization (D'Angelo et al. 2024).

From Parameter Space to Input Gradients. On the other hand, several studies identified a form of gradient pressure originating from the parameter space to the input space (Ma and Ying 2021; Dherin et al. 2022), indicating that flatter solutions tend to yield smoother decision boundaries.

Defenses against Black-Box Attacks

Adversarial training. While many defenses attempt to mitigate model evasion attacks with lightweight test-time input transformations (Guo et al. 2018) or auxiliary models (Nie et al. 2022), adversarial training remains the de facto standard. The basic idea is to augment the training data with adversarial examples or to use a fine-tuning phase, which adds a considerable computational complexity to this training-time defense. Here, it is crucial to ensure that the models do not overfit to the type of adversarial noise found in their training samples, but generalize to other (unseen) attacks (Tramèr et al. 2018). Techniques to further boost performance of adversarial training can, among others, leverage data augmentation (Rebuffi et al. 2021; Gowal et al. 2021) or topology alignment (Kuang et al. 2024).

Diverse ensemble strategies. A large body of works aims at improving the robustness of model ensembles as a whole, i.e., by ensuring that each ensemble member is dissimilar to the other members, making sure that an adversarial example only transfers to a few members (Yang et al. 2021, 2020; Pang et al. 2019; Deng and Mu 2023; Cai et al. 2023; Kariyappa and Qureshi 2019), e.g., by incorporating adversarial training with other ensemble members or by ensuring pairwise negative/orthogonal gradients.

3 Methodology

Preliminaries and Definitions

We denote the sample and label spaces with $\mathcal{X}$ and $\mathcal{Y}$, respectively, and the training data with $\mathcal{S} = (x_i, y_i)_{i=1}^{M}$, with $x_i \in \mathbb{R}^d, y_i \in \mathbb{R}^c$ and training set size M. We assume that $\mathcal{S}$ is sampled from the underlying data distribution $\mathcal{D}$.

A DNN-based classifier $\mathcal{F}_\theta : \mathcal{X} \to [0,1]^c$ is a function (parameterized by θ) that, given an input x, outputs the probability that the input is classified as each of the $n = |\mathcal{Y}|$ classes. The prediction of the classifier can be derived as $y = C(x) := \arg\max_{i \in [n]}(\mathcal{F}_\theta^i(x))$.

Adversarial examples. We define an adversarial example x' as a genuine image x to which carefully crafted adversarial noise is added, i.e., $x' = x + \delta$ for a small perturbation δ such that x' and x are perceptually indistinguishable to the human eye and yet are classified differently. Given a genuine input $x_0 \in \mathbb{R}^d$ predicted as $C(x_0) = s$ (source class), x' is an *adversarial example* of x_0 if $C(x') \neq s$ and $\|x' - x_0\|_p \leq \varepsilon$ for a given distortion bound $\varepsilon \in \mathbb{R}^+$ and l_p norm. The attacker searches for adversarial inputs x' with low distortion while maximizing a loss function $\mathcal{L}_\mathcal{F}$ (θ omitted for clarity), e.g., cross-entropy loss. Formally, the optimization problem is defined by: $\delta = \arg\max_{\|\delta\|_p \leq \varepsilon} \mathcal{L}_\mathcal{F}(x + \delta, y)$.

Definition 1 (Stochastic Gradient Descent)**.** SGD is a (noisy) optimization procedure for minimizing a loss function in training neural networks and is often combined with momentum and weight decay. For a minibatch of size B and weight decay λ, we define the gradient at iteration t as:

$$g_t = \frac{1}{B} \sum_{i=1}^{B} \nabla_\theta \mathcal{L}(\mathcal{F}_\theta(x_{i_t}), y_{i_t}) + \lambda \theta_t, \tag{1}$$

while drawing indices i.i.d. out of the uniform distribution $i_t \sim \mathbb{U}([N])$. The current velocity v weighs the velocity of the last iteration with momentum μ. The model parameters are then updated with learning rate η as follows:

$$v_t = \mu v_{t-1} + g_t, \quad \theta_{t+1} = \theta_t - \eta v_t. \tag{2}$$

Definition 2 (Model Smoothness)**.** Given a model $\mathcal{F}$ and a data distribution $\mathcal{D}$, the upper smoothness of $\mathcal{F}$ on $\mathcal{D}$ (Zhang et al. 2024) is defined as:

$$\overline{\sigma}_\mathcal{F} = \sup_{(x,y)\sim\mathcal{D}} \sigma(\nabla_x^2 \mathcal{L}_\mathcal{F}(x, y)), \tag{3}$$

where $\sigma(\cdot)$ denotes the largest eigenvalue, and $\nabla_x^2 \mathcal{L}_\mathcal{F}(x, y)$ the Hessian matrix computed w.r.t. x.

Definition 3 (Gradient Similarity)**.** Given two models $\mathcal{F}$ and $\mathcal{G}$ and their respective loss functions $\mathcal{L}_\mathcal{F}$ and $\mathcal{L}_\mathcal{G}$, we can compute the gradient similarity (Demontis et al. 2019; Zhang et al. 2024) based on the cosine similarity as follows:

$$\mathcal{S}(\mathcal{L}_\mathcal{F}, \mathcal{L}_\mathcal{G}, x, y) = \frac{\nabla_x \mathcal{L}_\mathcal{F}(x, y)^\intercal \nabla_x \mathcal{L}_\mathcal{G}(x, y)}{\|\nabla_x \mathcal{L}_\mathcal{F}(x, y)\|_2 \|\nabla_x \mathcal{L}_\mathcal{G}(x, y)\|_2} \tag{4}$$

and the upper gradient similarity based on the supremum as:

$$\overline{\mathcal{S}}(\mathcal{L}_\mathcal{F}, \mathcal{L}_\mathcal{G}) = \sup_{(x,y)\sim\mathcal{D}} \mathcal{S}(\mathcal{L}_\mathcal{F}, \mathcal{L}_\mathcal{G}, x, y) \tag{5}$$

Main Intuition

Hyperparameter selection is critical for training well-performing models and is a core component of any machine learning pipeline. Since hyperparameters induce implicit regularization, we investigate their impact on model robustness against evasion attacks in a black-box threat model.

In this work, we demonstrate that: (1) less smooth models are more robust to transfer-based attacks, as adversarial examples crafted on smoother surrogates are less likely to transfer to models with more variable loss landscapes (Demontis et al. 2019) (cf. Proposition 1); and (2) smoother models exhibit greater robustness against query-based attacks (Moosavi-Dezfooli et al. 2019). In Figure 2 (left), we see that adversarial examples generated towards a (typically smooth) surrogate model ($x_T'^{sur}$) transfer well to a smooth target model. However, the target remains robust to query-based attacks, as no adversarial example with perturbation magnitude $|\varepsilon|$ can be found within the query budget. In contrast, Figure 2 (right) shows that a less smooth model hinders transferability due to gradient misalignment but is more vulnerable to query-based attacks, which converge more easily.

Adapting the results from (Cohen et al. 2021; Kaur, Cohen, and Lipton 2022; Jastrzębski et al. 2018; D'Angelo et al. 2024; Ma and Ying 2021; Dherin et al. 2022), we relate the largest eigenvalue of the loss Hessian in parameter space to that in input space and obtain:

$$\sigma(\nabla_\theta^2 \mathcal{L}_\mathcal{F}(x, y)) \sim \sigma(\nabla_x^2 \mathcal{L}_\mathcal{F}(x, y)) \tag{6}$$

We empirically verify this connection in Figure 1 (left) and provide more details in the extended version.

1) Transferability & Model Smoothness. Let T_r denote the transferability for a benign sample x and an adversarial example x' which is generated on a surrogate model $\mathcal{F}$, but evaluated on a target model $\mathcal{G}$, as follows: $T_r(\mathcal{F}, \mathcal{G}, x) = \mathbb{1}[\mathcal{F}(x) = y \wedge \mathcal{G}(x) = y \wedge \mathcal{F}(x') \neq y \wedge \mathcal{G}(x') \neq y]$.

Proposition 1 (Less smooth models exhibit high robustness against transfer-based attacks)**.** *For a surrogate model $\mathcal{F}$ (with smoothness $\overline{\sigma}_\mathcal{F}$) and target model $\mathcal{G}$ (with smoothness $\overline{\sigma}_\mathcal{G}$), the upper bound on transferability* $\Pr((T_r(\mathcal{F}, \mathcal{G}, x)) = 1)$ *decreases when $\mathcal{G}$'s smoothness also decreases (i.e., $\overline{\sigma}_\mathcal{G}$ increases).*

In the following, we first determine smoothness and gradient similarity to be the main contributors to transferability, and then focus on how the regularization properties of hyperparameters reduce smoothness and gradient similarity and hence reduce the upper bound on transferability.

Based on (Yang et al. 2021; Zhang et al. 2024), we assume models $\mathcal{F}$ and $\mathcal{G}$ are $\overline{\sigma}_\mathcal{F}$-smooth and $\overline{\sigma}_\mathcal{G}$-smooth, respectively, with bounded gradient magnitude, i.e., $\|\nabla_x \mathcal{L}_\mathcal{F}(x, y)\| \leq B_\mathcal{F}$ and $\|\nabla_x \mathcal{L}_\mathcal{G}(x, y)\| \leq B_\mathcal{G}$ for any $x \in \mathcal{X}$, $y \in \mathcal{Y}$. We consider an untargeted attack with perturbation ball $\|\delta\|_2 \leq \varepsilon$. When the attack radius ε is small such that the denominator is larger than 0, the transferability can be upper bounded by:

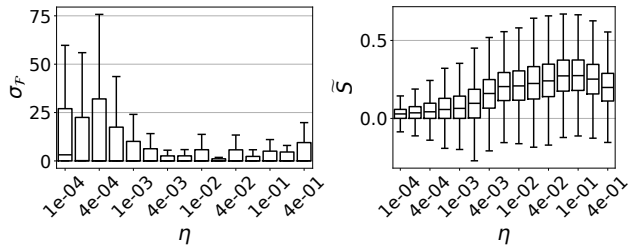


Figure 1: Impact of sharpness in parameter space (due to change of the learning rate hyperparameter) and smoothness in input space (left) and gradient similarity (right).

$$\Pr\left(T_r(\mathcal{F},\mathcal{G},x)=1\right) \leq \frac{\xi_{\mathcal{F}}}{\min\limits_{\substack{x\in\mathcal{X},y'\in\mathcal{Y}:\\(x,y)\in\text{supp}(\mathcal{D}),\\y'\neq y}} \mathcal{L}_{\mathcal{F}}(x,y') - \varepsilon B_{\mathcal{F}}\left(1+\sqrt{\frac{1+\overline{\mathcal{S}}(\mathcal{L}_{\mathcal{F}},\mathcal{L}_{\mathcal{G}})}{2}}\right) - \overline{\sigma}_{\mathcal{F}}\varepsilon^2} + \frac{\xi_{\mathcal{G}}}{\min\limits_{\substack{x\in\mathcal{X},y'\in\mathcal{Y}:\\(x,y)\in\text{supp}(\mathcal{D}),\\y'\neq y}} \mathcal{L}_{\mathcal{G}}(x,y') - \varepsilon B_{\mathcal{G}}\left(1+\sqrt{\frac{1+\overline{\mathcal{S}}(\mathcal{L}_{\mathcal{F}},\mathcal{L}_{\mathcal{G}})}{2}}\right) - \overline{\sigma}_{\mathcal{G}}\varepsilon^2}$$

Here $\xi_{\mathcal{F}}$ and $\xi_{\mathcal{G}}$ are the *empirical risks* of models $\mathcal{F}$ and $\mathcal{G}$, respectively, defined relative to a differentiable loss. The $\text{supp}(\mathcal{D})$ is the support of benign data distribution, i.e., x is the benign data and y is its associated true label. The full proof can be found in the extended version.

We observe that the surrogate model $\mathcal{F}$ and target model $\mathcal{G}$ equally contribute to this upper bound, with smoothness and gradient similarity being the main contributing properties.[1] Naturally, our aim is to reduce this upper bound to reduce the transferability of the model.

While the smoothness $\overline{\sigma}_{\mathcal{G}}$ (cf. Definition 2) (related to $B_{\mathcal{G}}$) is specific to the target, the gradient similarity $\overline{\mathcal{S}}(\mathcal{L}_{\mathcal{F}},\mathcal{L}_{\mathcal{G}})$ (cf. Definition 3) is a shared quantity with $\mathcal{F}$ and *positively correlated to the upper bound*. The relationship of regularization and alignment of input space gradients, i.e., their similarity, has been investigated by (Zhang et al. 2024) and found to be positively correlated, with varying impact depending on the used regularizer (which we empirically verify in Figure 1 (right) with more details in the extended version). As a result, the regularization properties of hyperparameters reduce smoothness and gradient similarity between the target and surrogate models, which in turn decreases the upper bound on transferability—thereby concluding this proposition.

2) Robustness against Query Attacks & Model Smoothness. The gradient of the loss surface in the input space, estimated by query-based attacks, is typically bounded by the true model gradient. More specifically, smoother models typically improve robustness by requiring a larger perturbation δ to successfully evade the model.

Namely, given a model $\mathcal{F}$, input x and label y, let

[1]This contrasts with prior work (Yang et al. 2021) that assumes that smoothness and gradient magnitude are identical for $\mathcal{F}$ and $\mathcal{G}$.

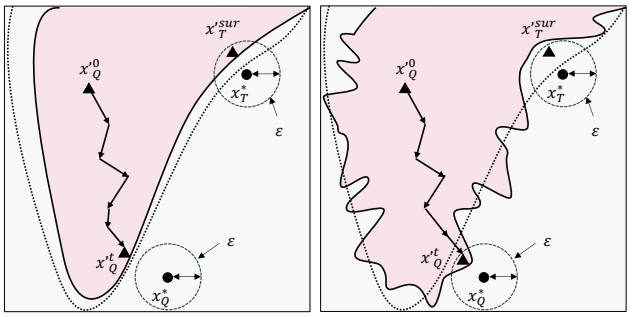


Figure 2: Tension between adversarial examples (▲) for transfer-based (x'_T) and query-based attacks (x'_Q) on a smooth (left) and a less smooth (right) model. The solid line and dotted line represent the decision boundaries for the target and surrogate model, respectively. The dashed line represents the ε constraint around a benign sample ($x^*_{T/Q}$ / ●).

$g = \nabla_x \mathcal{L}_{\mathcal{F}}(x,y)$ and $c := t - \mathcal{L}_{\mathcal{F}}(x,y) \geq 0$ with t as loss threshold. Further, we consider $\sigma_{\mathcal{F}} \geq 0$ as the largest eigenvalue of the Hessian matrix $\nabla^2_x \mathcal{L}_{\mathcal{F}}(x,y)$, and u_d as the respective eigenvector. For a perturbation δ, we now have the following bounds on the perturbation and hence robustness of a model as shown by (Moosavi-Dezfooli et al. 2019):

$$\frac{c}{\|g\|} - 2\sigma_{\mathcal{F}}\frac{c^2}{\|g\|^3} \leq \|\delta\| \leq \frac{c}{g^\intercal u_d} \tag{7}$$

Here, we see that smooth models with bounded (input) gradients exhibit higher robustness than less smooth ones. Concretely, increasing the curvature $\sigma_{\mathcal{F}}$, reduces the norm of δ, illustrating that smaller perturbations are sufficient for a successful adversarial example, while smaller curvature increases the required perturbation.

Interpretation. Based on Proposition 1 and our observations, we notice an inherent tension between choice of hyperparameters and robustness against transfer- and query-based attacks. Concretely, smoothness is beneficial for the robustness against query-based attacks (cf. Equation (7)) but detrimental for the robustness against transfer-based attacks (cf. Proposition 1), as this enables adversarial examples to transfer better from the typically smooth surrogates. The converse also holds: robustness against transfer-based attacks is improved with less smooth models and a decrease in gradient similarity, but weakened against query-based attacks by lowering the required perturbation magnitude.

Note that while the attacker has full control over the surrogate model $\mathcal{F}$, the target model $\mathcal{G}$ can be chosen arbitrarily by the defender. *We therefore propose using hyperparameter-induced implicit regularization to adjust model smoothness, enabling a trade-off between robustness to transfer-based and query-based attacks.*

4 Experimental Setup

Setup. All our experiments are run on an Ubuntu 24.04 machine with two NVIDIA A40 GPUs, two AMD EPYC 9554 64-core processors, and 512 GB of RAM. We used Python 3.11.11, CUDA 12.5, Lightning 2.5.0, and Ray Tune 2.40.0.

Parameter	Variable	Default	Range
Learning rate	η	0.1	$[0.0001, 0.4]$
Weight decay	λ	0.0005	$[0.000001, 0.01]$
Momentum	μ	0.9	$[0.8, 0.99]$
Batch size	B	128	$[32, 2048]$

Table 1: Typical SGD hyperparameters $\mathcal{H}$ for training a CIFAR-10 model and their value ranges in our experiments.

Attack Selection. We evaluate our approach against (i) a state-of-the-art transfer-attack, the Common Weakness attack (Chen et al. 2024), which uses an ensemble of surrogate models, and leverages sharpness aware minimization (SAM) to find solutions on a smooth area of a high loss region, and against (ii) a state-of-the-art query-based attack, the SquareAttack (Andriushchenko et al. 2020) from AutoAttack (Croce and Hein 2020).

We use a perturbation budget of $\varepsilon = 8/255$ under an l_∞ norm and generate adversarial examples for 1000 randomly sampled images from the test set for the transfer- and query-based attacks, respectively. For the transfer-based attacks, we consider an ensemble of 10 models trained with default parameters on various architectures. For the query-based attack, we assume a query budget of $Q = 500$.

Datasets & Models. We evaluate our approach on datasets of varying input dimensions and number of classes, i.e., CIFAR-10 (Krizhevsky 2009) and ImageNet (Russakovsky et al. 2015) datasets. The former contains $50,000$ train and $10,000$ test images of size 32×32 pixels, divided into 10 classes. The latter contains 1.2 million training images and a validation set of $50,000$ images. For both datasets, we focus on models of the ResNet family (He et al. 2016) due to their wide adoption. We include additional results on MobileNetV2 in the extended version.

In addition to a basic centralized model training on the full dataset, we also consider various ensemble instantiations that use logit averaging (before softmax) for inference:

- Deep Ensemble (full dataset, different initialization)
- Distributed IID Ensemble (disjoint dataset with i.i.d. data)
- Distributed Non-IID Ensemble (disjoint dataset with non-i.i.d. data using Dirichlet distribution with $\alpha = 0.9$)

Metrics—(Robust) accuracy. Let $\mathcal{S} \subset \mathcal{X} \times \mathcal{Y}$ denote the set of genuine samples provided to the attacker $\mathcal{A}$, and let ε denote the distortion budget. To compute the robust accuracy (RA), we measure the accuracy on the adversarial examples, i.e., $\|x' - x\|_p \leq \varepsilon$, generated by the attacker:

$$\mathsf{RA} := |\{(x,x') \in \mathcal{X} \times \mathcal{A}(\mathcal{S}) \mid C(x) = C(x')\}| / |\mathcal{S}|, \tag{8}$$

where $\mathcal{A}(\mathcal{S})$ denotes the set of candidate adversarial examples output by $\mathcal{A}$ in a run of the attack on input $\mathcal{S}$. In contrast to the clean accuracy (CA) computed on benign images, RA is computed on adversarial examples. The complement of RA is the attack success rate (ASR), i.e., $\mathsf{ASR} = 1 - \mathsf{RA}$.

Pareto frontier. The Pareto frontier emerges as an effective tool to evaluate tradeoffs in multi-objective optimization. In our case, between CA and RA.

A solution ω^* is *Pareto optimal* if there exists no other solution that improves all objectives simultaneously. Formally, given two solutions ω_1 and ω_2, we write $\omega_1 \succ \omega_2$ if ω_1 dominates ω_2, i.e., if $\mathsf{CA}(\omega_1) \geq \mathsf{CA}(\omega_2) \wedge \mathsf{RA}(\omega_1) \geq \mathsf{RA}(\omega_2)$, where $\mathsf{CA}(\omega)$ and $\mathsf{RA}(\omega)$ denote the clean accuracy and robust accuracy as functions of the parameter ω. The *Pareto frontier* is the set of Pareto-optimal solutions:

$$PF(\Omega) = \{\omega^* \in \Omega \mid \nexists \omega \in \Omega \text{ s.t. } \omega \succ \omega^*\}.$$

Algorithm 1: Ablation of hyperparameters

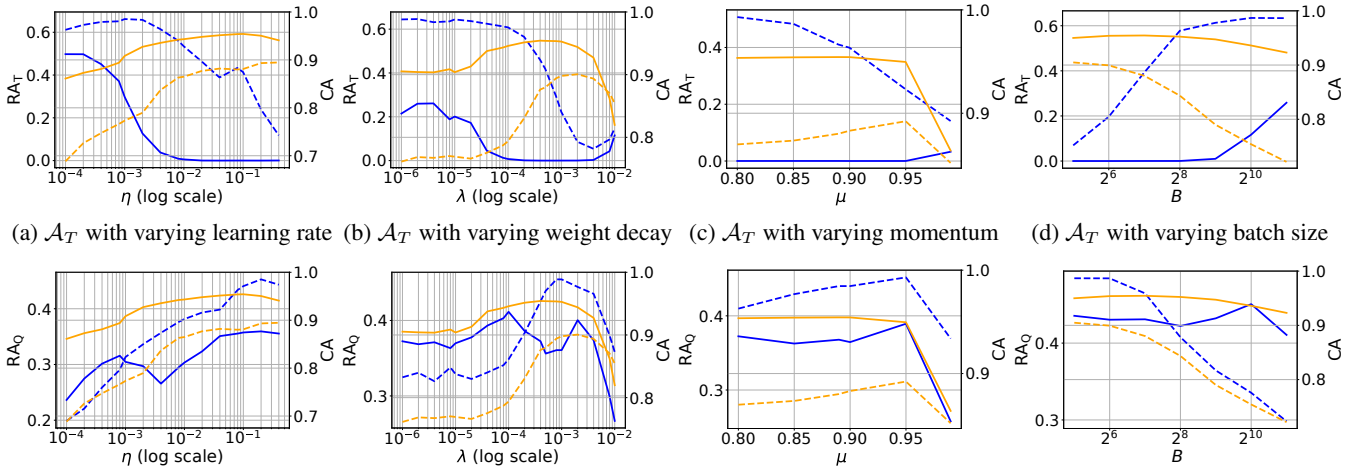
Input: dataset `data`, ablation ranges `ablation_ranges`
Output: All ablated models `ablated_models`

```
/* For each type of hyperparameter, we
vary it in the given range while keeping
the others fixed (cf. Table 1)  */
ablated_models = [[], [], [], []]
default_params = [0.1, 0.0005, 0.9, 128]
for param_pos in range(len(['learning_rate', 'weight_decay',
'momentum', 'batch_size'])) do
    param_models = []
    params = default_params.copy()
    for param_value in ablation_ranges[param_pos] do
        params[param_pos] = param_value
        model = train_model(params, data)
        param_models.append(model)
    end for
    ablated_models[param_pos] = param_models
end for
return ablated_models
```

5 Experimental Results

To answer RQ1 and RQ2, i.e., understand the impact of hyperparameters on the robustness provisions of transfer- (RA_T) and query-based (RA_Q) attacks, based on the concrete attacks introduced in Section 4, respectively, we focus on the four training hyperparameters that are found in a typical model training using SGD (cf. Equation (1) and Equation (2)). The considered range of hyperparameters can be found in Table 1. For these, we fix all but one parameter, vary it across a range of values, and monitor the impact on CA and RA of all considered attacks (cf. Algorithm 1). We average the data points that we obtain across the number of nodes N (each data point is averaged over three independent runs), encompassing centralized deployments ($N = 1$), distributed deployments with ($N = 3, 5, 7$), and data distributions for distributed ML, as deviations originating from these parameters/setups are limited. The impact of N and data distribution is discussed in an ablation study in Section 6. To answer RQ3, we perform a hyperparameter search (over 100 configurations) using the efficient NSGA-II genetic algorithm (Deb et al. 2002) adapted from Optuna and integrate it in our framework via its Ray integration.

Our default configuration for all experiments consists of a ResNet-18 trained with SGD for 200 epochs (with early stopping) and a CosineAnnealing learning rate scheduler on the CIFAR-10 dataset (with 20% validation data).



(a) $\mathcal{A}_T$ with varying learning rate (b) $\mathcal{A}_T$ with varying weight decay (c) $\mathcal{A}_T$ with varying momentum (d) $\mathcal{A}_T$ with varying batch size

(e) $\mathcal{A}_Q$ with varying learning rate (f) $\mathcal{A}_Q$ with varying weight decay (g) $\mathcal{A}_Q$ with varying momentum (h) $\mathcal{A}_Q$ with varying batch size

Figure 3: Robust accuracy for all hyperparameters $\mathcal{H} = (\eta, \lambda, \mu, B)$ for transfer-based ($\mathcal{A}_T$) and query-based attackers ($\mathcal{A}_Q$). Our results are averaged over all nodes for deep ensembles (solid-line) and distributed ML (dashed-line) on CIFAR-10. The blue line shows RA, while the orange line shows CA. Each column varies the specified hyperparameter, while fixing all others.

RQ 1: As shown in Figure 3 (first row), we observe a consistent improvement in RA of up to 55% against transfer-based attacks across multiple ML instantiations when the hyperparameters decrease (or batch size B increases). This matches our observations from Equation (6) and Proposition 1.

When considering the impact of the choice of the *learning rate* on deep ensembles, we observe that a reduction of η starts to improve RA at $\eta = 0.01$, reaching RA $= 0.5$ for $\eta = 0.0001$. At the same time, CA maintains an excellent performance of as low as 86%. In contrast, the distributed ML instantiations obtain a notably different CA/RA behavior compared to deep ensembles. Here, we see a consistently higher RA across the entire spectrum of η, already maintaining a RA $= 42\%$ at the default $\eta = 0.1$ and improving it further to RA $= 67\%$ at $\eta = 0.001$. Due to the disjoint split data in distributed ML, we see a steeper decline in CA, reaching a still competitive CA $= 0.78$ at the highest RA. Concretely, we observe the largest robustness improvement of up to 64% for the i.i.d. distributed ML instance with $N = 3$ and $\eta = 0.001$.

For *weight decay*, we see a similar trend, i.e., an improvement in RA with a decreasing λ, as this regularizes the model less. We monitor a RA of only around 27% for deep ensembles, while the distributed ML instantiations reach a RA $= 65\%$ at CA $= 77\%$ for $\lambda =$ 2e-6 (similar to learning rate). The *momentum* hyperparameter continues the previous trend, yet improvements for RA for deep ensembles are almost non-existent at a maximum RA $= 4\%$. In contrast, distributed ML can reach a RA $= 51\%$ and CA $= 87\%$ at $\mu = 0.8$. For *batch size*, we start to see an improvement in RA only at $B = 2048$ with RA $= 26\%$. For distributed ML, we observe a RA as high as 64% at the highest B of 2048.

Takeaway 1. Reducing the value of η, λ, μ and increasing B improves RA_T with minimal impact on CA.

Overall, distributed ML consistently yields higher RA_T than deep ensembles trained on the full dataset, with a slight impact on CA. This stems from increased ensemble heterogeneity—due to disjoint data and reduced smoothness (as models converge toward sharper minima)—which lowers gradient similarity between the ensemble members (cf. Proposition 1).

RQ 2: Conforming with our analysis in Equations (6) and (7), and in contrast to transfer-based attacks, we observe a consistent impact on RA in Figure 3 with improvements of up to 28% as η, λ increase and μ, B decrease.

We observe that a reduction of η reduces RA. For deep ensembles, we obtain the highest RA $= 36\%$ at $\eta = 0.2$ and see a decline in RA until $\eta = 0.004$ at a RA $= 27\%$, after which we see a slight increase to RA $= 32\%$ at $\eta = 0.0008$. Afterwards, RA drops as low as 24%. In contrast, we observe a smoother relationship between η and RA in distributed ML, achieving the highest RA $= 0.46$ at $\eta = 0.2$.

For *weight decay*, we monitor a relatively constant RA for deep ensembles, fluctuating between 36% and 42% across the spectrum of λ until $\lambda = 0.008$ with a reduction on CA of at most 5%. For distributed ML, we obtain a larger range of RA between 32% and 46% at higher fluctuations in CA between 76% and 90%. For *momentum*, we see a constant CA/RA across the entire range of μ for both deep ensembles, and distributed ML, with a slight decrease in both CA/RA for $\mu = 0.99$. For *batch size*, the previously observed trend continues: deep ensembles remain at a consistent CA/RA across the entire hyperparameter range, while distributed ML follows the behavior found with η, albeit inverted, due to the scaling nature of batch size on learning rate.

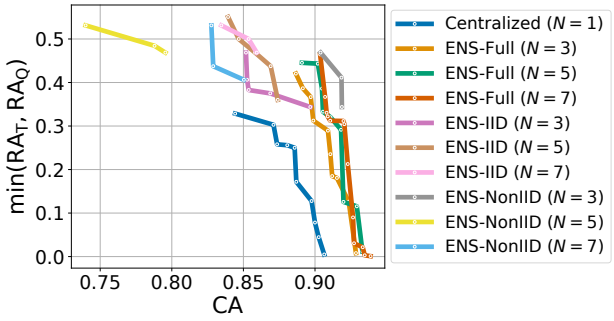


Figure 4: Pareto frontier of the NSGA-II search across all ML instantiations and hyperparameters.

Instance		Hyperparameter tuning $\mathcal{H}$			JPEG	Adv. Training
		Avg.	Ours	Δ	Δ	Δ
Central.	CA	0.92	0.87	−0.05	−0.02	−0.01
	RA$_T$	0.08	0.51	**+0.43**	−0.16	+0.18
	RA$_Q$	0.28	0.30	**+0.02**	+0.00	+0.38
ENS-Full	CA	0.94	0.90	−0.04	−0.03	−0.04
	RA$_T$	0.07	0.47	**+0.40**	−0.10	+0.22
	RA$_Q$	0.42	0.46	**+0.04**	−0.08	+0.22
ENS-IID	CA	0.86	0.87	**+0.01**	−0.02	−0.01
	RA$_T$	0.43	0.62	**+0.19**	+0.01	+0.07
	RA$_Q$	0.39	0.44	**+0.05**	−0.02	+0.24
ENS-Non-IID	CA	0.86	0.90	**+0.04**	−0.05	−0.04
	RA$_T$	0.32	0.51	**+0.19**	+0.05	+0.18
	RA$_Q$	0.37	0.47	**+0.10**	−0.07	+0.21

Table 2: Comparison of CA, RA of the recommended $\mathcal{H}$ found by NSGA-II vs. the default parameters and defenses.

Takeaway 2. We find that an increase in η, λ enhances RA$_Q$, while an inverse relationship holds for μ, B. Deep ensembles show more stable RA$_Q$, likely due to access to the full dataset, leading to flatter minima and smoother input spaces. Distributed ML benefits from this effect especially for larger η and λ.

RQ 3: Given RQ1 and RQ2, we now investigate whether it is possible to tune hyperparameters to achieve strong robustness against both types of black-box attacks. To this end, similar to (Lachnit and Karame 2025; Bagdasarian and Shmatikov 2024), we make use of the NSGA-II genetic hyperparameter optimization algorithm (Deb et al. 2002) to identify optimal hyperparameter configurations while producing a diverse Pareto front. Specifically, we use the Optuna library with the NSGA-II sampler, integrated into our framework via Ray. Our objective is to find the hyperparameter configuration $\mathcal{H}$ that simultaneously maximizes both robustness criteria, formalized as follows: $\arg\max_{\mathcal{H}}(\text{CA}^{\mathcal{H}}, \min(\text{RA}_\text{T}^{\mathcal{H}}, \text{RA}_\text{Q}^{\mathcal{H}}))$.

This approach allows us to identify hyperparameter configurations that (1) maintain strong performance on benign data and (2) guarantee a minimum level of RA against both types of black-box attacks. We run NSGA-II with a population size of 20 and 5 generations, yielding a total of 100 distinct hyperparameter combinations. Figure 4 presents the results, illustrating the Pareto fronts for all ML instances.

We observe the following general trends: deep ensembles significantly enhance both CA and RA as the number of nodes increases, achieving a favorable trade-off at CA $= 0.90$ and RA $= 0.46$ for $N = 7$. However, performance gains tend to saturate beyond $N = 3$. In the distributed ML setting with i.i.d. data, results are closely clustered: $N = 5$ achieves the highest RA of 55%, while $N = 3$ yields the best CA at 90%. For non-i.i.d. data, the results vary more widely, as the distribution of class splits across nodes significantly impacts performance. Notably, $N = 5$ performs the worst in terms of CA, remaining below 80%. The best Pareto front is observed at $N = 3$.

We now analyze the best-performing configurations for each ML paradigm ($N = 7$ for deep ensembles, and $N = 5$ and $N = 3$ for distributed ML with i.i.d. and non-i.i.d. data, respectively), focusing on CA, RA$_T$, and RA$_Q$. We select the best RA values from the hyperparameter search, allowing up to 5% decrease in CA relative to the centralized ($N = 1$) baseline. In Table 2, we compare the results of the best $\mathcal{H}$ configurations (see extended version) to the average CA, RA$_T$, and RA$_Q$ obtained across all $\mathcal{H}$ combinations from RQ1 and RQ2 (see extended version). Overall, we see consistent improvements across all ML paradigms—up to 43% in RA$_T$, 10% in RA$_Q$, and 4% in CA.

Comparison with SOTA defenses: Our recommended hyperparameters significantly improve robustness over JPEG (Guo et al. 2018) by up to 16% while exhibiting 78% fewer epochs (compared to the default settings in Table 1, reducing from an average of 186 down to just 41 epochs) on the same architecture.[2] This represents a significant performance gain, especially compared to adversarial training (Rebuffi et al. 2021; Gowal et al. 2021), which can require up to 40 GPU hours per model for an average increase of 21% in robustness over our approach.

Takeaway 3. Compared to SOTA defenses like JPEG (Guo et al. 2018) and adversarial training (Gowal et al. 2021), our results in Table 2 demonstrate that our approach (especially in the distributed ML instance) strikes a strong efficiency tradeoff.

6 Ablation Study

We now ablate our results w.r.t. (1) the number of nodes, (2) data distribution, and (3) complex datasets (e.g., ImageNet). We focus on the learning rate η, as results for other hyperparameters—provided in the extended version—show similar trends.

Impact of N. As shown in Table 3, we observe that higher node counts can achieve up to a 4% improvement in CA, due to their diverse initialization and hence better generalization. While the improvements in RA$_T$ are limited to up to 7%, RA$_Q$ improves by up to 20% for the lower learning rates, which we relate to a smoothing of the loss landscape due to the ensembling of diverse, less smooth models.

[2]This is due to faster convergence to a well-generalizing minimum, which also leads to earlier onset of overfitting.

Instance	N		Learning rate η														
			1e-4	2e-4	4e-4	8e-4	1e-3	2e-3	4e-3	8e-3	0.01	0.02	0.04	0.08	0.1	0.2	0.4
Central.	1	CA	0.83	0.84	0.85	0.87	0.89	0.91	0.92	0.93	0.93	0.94	0.94	0.95	0.95	0.94	0.93
		RA_T	0.48	0.49	0.41	0.39	0.31	0.15	0.05	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00
		RA_Q	0.13	0.15	0.17	0.21	0.20	0.20	0.21	0.23	0.22	0.22	0.26	0.29	0.26	0.26	0.26
ENS-Full	3	CA	0.87	0.88	0.88	0.90	0.91	0.93	0.94	0.94	0.94	0.95	0.95	0.96	0.95	0.95	0.94
		RA_T	0.48	0.48	0.45	0.35	0.32	0.13	0.04	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		RA_Q	0.26	0.29	0.32	0.33	0.32	0.31	0.25	0.29	0.33	0.33	0.34	0.38	0.37	0.39	0.38
	5	CA	0.87	0.89	0.89	0.90	0.92	0.93	0.94	0.95	0.95	0.95	0.95	0.96	0.96	0.95	0.95
		RA_T	0.51	0.51	0.47	0.37	0.28	0.12	0.04	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00
		RA_Q	0.28	0.30	0.36	0.34	0.35	0.34	0.30	0.32	0.32	0.37	0.40	0.39	0.40	0.40	0.38
	7	CA	0.87	0.88	0.89	0.90	0.92	0.93	0.94	0.95	0.95	0.95	0.95	0.96	0.96	0.95	0.95
		RA_T	0.52	0.52	0.48	0.38	0.28	0.12	0.03	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		RA_Q	0.28	0.35	0.35	0.39	0.35	0.34	0.31	0.34	0.35	0.37	0.41	0.37	0.40	0.39	0.41
ENS-IID	3	CA	0.75	0.78	0.81	0.82	0.83	0.85	0.89	0.90	0.90	0.91	0.92	0.92	0.92	0.92	0.92
		RA_T	0.62	0.65	0.66	0.64	0.66	0.63	0.49	0.38	0.30	0.20	0.10	0.12	0.07	0.02	0.02
		RA_Q	0.20	0.22	0.26	0.30	0.32	0.35	0.35	0.35	0.36	0.39	0.42	0.46	0.48	0.46	0.44
	5	CA	0.69	0.73	0.75	0.77	0.78	0.80	0.84	0.87	0.88	0.89	0.89	0.89	0.89	0.90	0.90
		RA_T	0.62	0.65	0.67	0.68	0.69	0.68	0.66	0.59	0.57	0.48	0.39	0.48	0.45	0.14	0.06
		RA_Q	0.21	0.24	0.25	0.30	0.30	0.36	0.36	0.39	0.40	0.37	0.38	0.47	0.45	0.45	0.44
	7	CA	0.65	0.70	0.72	0.73	0.74	0.77	0.81	0.84	0.85	0.86	0.87	0.85	0.86	0.88	0.89
		RA_T	0.61	0.63	0.65	0.67	0.67	0.70	0.70	0.67	0.67	0.62	0.57	0.63	0.63	0.47	0.17
		RA_Q	0.20	0.23	0.28	0.30	0.29	0.31	0.38	0.39	0.39	0.42	0.42	0.39	0.41	0.46	0.48
ENS-Non-IID	3	CA	0.73	0.76	0.78	0.80	0.81	0.81	0.87	0.88	0.87	0.88	0.89	0.90	0.90	0.90	0.90
		RA_T	0.61	0.62	0.63	0.63	0.63	0.61	0.48	0.45	0.42	0.34	0.21	0.22	0.21	0.03	0.02
		RA_Q	0.16	0.18	0.22	0.27	0.31	0.31	0.31	0.35	0.34	0.38	0.36	0.42	0.42	0.42	0.39
	5	CA	0.68	0.72	0.74	0.76	0.76	0.79	0.82	0.87	0.87	0.87	0.88	0.88	0.88	0.89	0.89
		RA_T	0.61	0.64	0.65	0.65	0.66	0.68	0.68	0.60	0.56	0.51	0.44	0.49	0.47	0.26	0.14
		RA_Q	0.21	0.23	0.27	0.30	0.33	0.36	0.36	0.39	0.40	0.39	0.40	0.46	0.48	0.46	0.42
	7	CA	0.63	0.67	0.69	0.70	0.71	0.72	0.78	0.81	0.81	0.85	0.84	0.83	0.84	0.86	0.87
		RA_T	0.60	0.62	0.64	0.64	0.66	0.66	0.68	0.67	0.68	0.63	0.61	0.65	0.65	0.52	0.29
		RA_Q	0.22	0.21	0.28	0.27	0.33	0.34	0.39	0.38	0.40	0.41	0.41	0.40	0.41	0.47	0.48

Table 3: CA and RA of across ML instantiations, and number of nodes using the learning rate hyperparameter η.

	Learning rate η								
	1e-3	2e-3	4e-3	8e-3	0.01	0.04	0.1	0.2	0.4
CA	0.47	0.56	0.61	0.64	0.65	0.68	0.69	0.68	0.67
RA_T	0.20	0.21	0.18	0.16	0.14	0.11	0.11	0.09	0.11
RA_Q	0.12	0.23	0.24	0.30	0.30	0.31	0.34	0.33	0.31

Table 4: CA and RA on ImageNet using hyperparameter η.

For distributed ML, we generally observe a steeper decline in CA with an increasing number of nodes, as the total amount of data for each weak learner is reduced. While improvements in RA_Q are limited to at most 8%, RA_T strongly improves for $\eta > 0.01$, concretely by up to 56% for $\eta = 0.1$ in the i.i.d. setup when increasing the node count from 3 to 7. The diversity within the ensemble is increasing with the number of nodes, making it more difficult for adversarial examples to transfer successfully. Both of these results align with our observations while answering RQ3.

Impact of Data Distribution. With respect to the data distribution, we observe that the use of i.i.d. data performs slightly better (by up to 4%) in CA than the use of non-i.i.d. data. In general, we see deviations in RA_T of up to 12% and RA_Q of up to 6%. In contrast, we observe an interesting behavior when comparing $N = 3$ for $\eta > 0.01$, where the most significant improvements are seen for non-i.i.d. data. Concretely, we see an improvement in RA_T by up to 21%, with a decrease in RA_Q by 8% at a reduction in CA by 3% (confirming our results in RQ3).

Impact of Dataset. To understand the impact of the dataset on our approach, we adopt our setup used for answering RQ 1 and RQ 2, and train a model on the ImageNet dataset for 90 epochs with a StepLR learning rate scheduler (following the official PyTorch parameters). Due to the larger dataset and longer training times, we focus only on analyzing the impact of the learning rate and vary it across multiple training runs. As shown in Table 4, our findings based on CIFAR-10 also hold for ImageNet. Concretely, we see that a reduction in η improves RA_T, while decreasing RA_Q.

7 Conclusion

In this paper, we demonstrate a striking contrast in how training hyperparameters affect robustness under distinct black-box threat models. Specifically, we find that decreasing the learning rate substantially improves robustness against transfer-based attacks, by up to 64% across various ML instantiations. In contrast, for query-based attacks, we observe that increasing the learning rate improves robustness by up to 28%. Supported by thorough experiments, we identify configurations that strike an effective balance—enhancing robustness against both transfer-based and query-based attacks simultaneously. We therefore hope that our findings motivate further research in this fascinating area.

Acknowledgments

This work has been co-funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2092 CASA - 390781972 and by the German Federal Ministry of Research, Technology and Space (BMFTR) through the project TRAIN (01IS23027A).

References

Andreina, S.; Zimmer, P.; and Karame, G. 2025. On the Robustness of Distributed Machine Learning against Transfer Attacks. In Walsh, T.; Shah, J.; and Kolter, Z., eds., *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, 15382–15390. AAAI Press.

Andriushchenko, M.; Croce, F.; Flammarion, N.; and Hein, M. 2020. Square Attack: A Query-Efficient Black-Box Adversarial Attack via Random Search. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII*, 484–501. Berlin, Heidelberg: Springer-Verlag. ISBN 978-3-030-58591-4.

Bagdasarian, E.; and Shmatikov, V. 2024. Mithridates: Auditing and Boosting Backdoor Resistance of Machine Learning Pipelines. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, 4480–4494. Salt Lake City UT USA: ACM. ISBN 979-8-4007-0636-3.

Biggio, B.; Corona, I.; Maiorca, D.; Nelson, B.; Srndic, N.; Laskov, P.; Giacinto, G.; and Roli, F. 2013. Evasion Attacks against Machine Learning at Test Time. In Blockeel, H.; Kersting, K.; Nijssen, S.; and Zelezný, F., eds., *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III*, volume 8190 of *Lecture Notes in Computer Science*, 387–402. Springer.

Cai, Y.; Ning, X.; Yang, H.; and Wang, Y. 2023. Ensemble-in-One: Ensemble Learning within Random Gated Networks for Enhanced Adversarial Robustness. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(12): 14738–14747.

Chen, H.; Zhang, Y.; Dong, Y.; Yang, X.; Su, H.; and Zhu, J. 2024. Rethinking Model Ensemble in Transfer-Based Adversarial Attacks. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Chen, J.; Jordan, M. I.; and Wainwright, M. J. 2020. HopSkipJumpAttack: A Query-Efficient Decision-Based Attack. In *2020 IEEE Symposium on Security and Privacy (SP)*, 1277–1294. San Francisco, CA, USA: IEEE. ISBN 978-1-7281-3497-0.

Chen, P.-Y.; Zhang, H.; Sharma, Y.; Yi, J.; and Hsieh, C.-J. 2017. ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 15–26. Dallas Texas USA: ACM. ISBN 978-1-4503-5202-4.

Cohen, J.; Kaur, S.; Li, Y.; Kolter, J. Z.; and Talwalkar, A. 2021. Gradient Descent on Neural Networks Typically Occurs at the Edge of Stability. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Croce, F.; and Hein, M. 2020. Reliable Evaluation of Adversarial Robustness with an Ensemble of Diverse Parameter-Free Attacks. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, 2206–2216. PMLR.

D'Angelo, F.; Andriushchenko, M.; Varre, A. V.; and Flammarion, N. 2024. Why Do We Need Weight Decay in Modern Deep Learning? In Globersons, A.; Mackey, L.; Belgrave, D.; Fan, A.; Paquet, U.; Tomczak, J. M.; and Zhang, C., eds., *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Deb, K.; Pratap, A.; Agarwal, S.; and Meyarivan, T. 2002. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2): 182–197.

Demontis, A.; Melis, M.; Pintor, M.; Jagielski, M.; Biggio, B.; Oprea, A.; Nita-Rotaru, C.; and Roli, F. 2019. Why Do Adversarial Attacks Transfer? Explaining Transferability of Evasion and Poisoning Attacks. In *28th USENIX Security Symposium (USENIX Security 19)*, 321–338.

Deng, Y.; and Mu, T. 2023. Understanding and Improving Ensemble Adversarial Defense. In Oh, A.; Naumann, T.; Globerson, A.; Saenko, K.; Hardt, M.; and Levine, S., eds., *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Dherin, B.; Munn, M.; Rosca, M.; and Barrett, D. 2022. Why Neural Networks Find Simple Solutions: The Many Regularizers of Geometric Complexity. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting Adversarial Attacks with Momentum. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 9185–9193. Computer Vision Foundation / IEEE Computer Society.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and Harnessing Adversarial Examples. In Bengio, Y.; and LeCun, Y., eds., *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Gowal, S.; Qin, C.; Uesato, J.; Mann, T.; and Kohli, P. 2021. Uncovering the Limits of Adversarial Training against Norm-Bounded Adversarial Examples. arXiv:2010.03593.

Guo, C.; Rana, M.; Cissé, M.; and van der Maaten, L. 2018. Countering Adversarial Images Using Input Transformations. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Confer-*

ence on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, 770–778. IEEE Computer Society.

Ilyas, A.; Engstrom, L.; Athalye, A.; and Lin, J. 2018. Black-Box Adversarial Attacks with Limited Queries and Information. In Dy, J. G.; and Krause, A., eds., *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, 2142–2151. PMLR.

Jastrzębski, S.; Kenton, Z.; Arpit, D.; Ballas, N.; Fischer, A.; Bengio, Y.; and Storkey, A. 2018. Three Factors Influencing Minima in SGD. arXiv:1711.04623.

Kariyappa, S.; and Qureshi, M. K. 2019. Improving Adversarial Robustness of Ensembles with Diversity Training. arXiv:1901.09981.

Kaur, S.; Cohen, J.; and Lipton, Z. C. 2022. On the Maximum Hessian Eigenvalue and Generalization. In Antorán, J.; Blaas, A.; Feng, F.; Ghalebikesabi, S.; Mason, I.; Pradier, M. F.; Rohde, D.; Ruiz, F. J. R.; and Schein, A., eds., *Proceedings on "I Can't Believe It's Not Better! - Understanding Deep Learning through Empirical Falsification" at NeurIPS 2022 Workshops, 03 December 2022, New Orleans, Louisiana, USA*, volume 187 of *Proceedings of Machine Learning Research*, 51–65. PMLR.

Krizhevsky, A. 2009. Learning Multiple Layers of Features from Tiny Images. Technical report.

Kuang, H.; Liu, H.; Lin, X.; and Ji, R. 2024. Defense Against Adversarial Attacks Using Topology Aligning Adversarial Training. *IEEE Transactions on Information Forensics and Security*, 19: 3659–3673.

Lachnit, S.; and Karame, G. 2025. On Hyperparameters and Backdoor-Resistance in Horizontal Federated Learning. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, CCS '25, 1919–1933. New York, NY, USA: Association for Computing Machinery. ISBN 9798400715259.

Ma, C.; and Ying, L. 2021. On Linear Stability of SGD and Input-Smoothness of Neural Networks. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*, volume 34, 16805–16817. Curran Associates, Inc.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; Uesato, J.; and Frossard, P. 2019. Robustness via Curvature Regularization, and Vice Versa. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9070–9078. Long Beach, CA, USA: IEEE. ISBN 978-1-7281-3293-8.

Nie, W.; Guo, B.; Huang, Y.; Xiao, C.; Vahdat, A.; and Anandkumar, A. 2022. Diffusion Models for Adversarial Purification. In Chaudhuri, K.; Jegelka, S.; Song, L.; Szepesvári, C.; Niu, G.; and Sabato, S., eds., *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, 16805–16827. PMLR.

Pang, T.; Xu, K.; Du, C.; Chen, N.; and Zhu, J. 2019. Improving Adversarial Robustness via Promoting Ensemble Diversity. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, 4970–4979. PMLR.

Rebuffi, S.-A.; Gowal, S.; Calian, D. A.; Stimberg, F.; Wiles, O.; and Mann, T. 2021. Fixing Data Augmentation to Improve Adversarial Robustness. arXiv:2103.01946.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A. C.; and Fei-Fei, L. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3): 211–252.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I. J.; and Fergus, R. 2014. Intriguing Properties of Neural Networks. In Bengio, Y.; and LeCun, Y., eds., *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I. J.; Boneh, D.; and McDaniel, P. D. 2018. Ensemble Adversarial Training: Attacks and Defenses. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Tu, C.-C.; Ting, P.; Chen, P.-Y.; Liu, S.; Zhang, H.; Yi, J.; Hsieh, C.-J.; and Cheng, S.-M. 2019. AutoZOOM: Autoencoder-Based Zeroth Order Optimization Method for Attacking Black-Box Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33: 742–749.

Wu, W.; Su, Y.; Lyu, M. R.; and King, I. 2021. Improving the Transferability of Adversarial Samples with Adversarial Transformations. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, Virtual, June 19-25, 2021*, 9024–9033. Computer Vision Foundation / IEEE.

Yang, H.; Zhang, J.; Dong, H.; Inkawhich, N.; Gardner, A.; Touchet, A.; Wilkes, W.; Berry, H.; and Li, H. 2020. DVERGE: Diversifying Vulnerabilities for Enhanced Robust Generation of Ensembles. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.-F.; and Lin, H.-T., eds., *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, Virtual*.

Yang, Z.; Li, L.; Xu, X.; Zuo, S.; Chen, Q.; Zhou, P.; Rubinstein, B. I. P.; Zhang, C.; and Li, B. 2021. TRS: Transferability Reduced Ensemble via Promoting Gradient Diversity and Model Smoothness. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y. N.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, Virtual*, 17642–17655.

Zhang, Y.; Hu, S.; Zhang, L. Y.; Shi, J.; Li, M.; Liu, X.; Wan, W.; and Jin, H. 2024. Why Does Little Robustness Help? A Further Step Towards Understanding Adversarial Transferability. In *2024 IEEE Symposium on Security and Privacy (SP)*, 3365–3384. San Francisco, CA, USA: IEEE. ISBN 979-8-3503-3130-1.