Artificial Intelligence Review (2023) 56:S217–S251
https://doi.org/10.1007/s10462-023-10521-4

Safety-critical computer vision: an empirical survey of adversarial evasion attacks and defenses on computer vision systems

Charles Meyers[1] · Tommy Löfstedt[1] · Erik Elmroth[1]

Accepted: 28 May 2023 / Published online: 23 June 2023

Abstract
Considering the growing prominence of production-level AI and the threat of adversarial attacks that can poison a machine learning model against a certain label, evade classification, or reveal sensitive data about the model and training data to an attacker, adversaries pose fundamental problems to machine learning systems. Furthermore, much research has focused on the inverse relationship between robustness and accuracy, raising problems for real-time and safety-critical systems particularly since they are governed by legal constraints in which software changes must be explainable and every change must be thoroughly tested. While many defenses have been proposed, they are often computationally expensive and tend to reduce model accuracy. We have therefore conducted a large survey of attacks and defenses and present a simple and practical framework for analyzing any machine-learning system from a safety-critical perspective using adversarial noise to find the upper bound of the failure rate. Using this method, we conclude that all tested configurations of the ResNet architecture fail to meet any reasonable definition of 'safety-critical' when tested on even small-scale benchmark data. We examine state of the art defenses and attacks against computer vision systems with a focus on safety-critical applications in autonomous driving, industrial control, and healthcare. By testing a combination of attacks and defenses, their efficacy, and their run-time requirements, we provide substantial empirical evidence that modern neural networks consistently fail to meet established safety-critical standards by a wide margin.

Keywords Adversarial machine learning · Computer vision · Autonomous vehicles · Safety-critical

✉ Charles Meyers
cmeyers@cs.umu.se

Tommy Löfstedt
tommy@cs.umu.se

Erik Elmroth
elmroth@cs.umu.se

1 Department of Computing Science, Umeå University, Umeå, Sweden

1 Introduction

Vehicular accidents, medical mistakes, and industrial safety failures are among the leading causes of preventable death around the world (The Organisation for Economic Co-operation and Development 2020; Makary and Daniel 2016; ICOH 2017). Technologies like image classification systems have shown to be more accurate than their human counterparts under strict laboratory conditions in these domains (NHTSA 2015; Pakdemirli 2019; Bernal et al. 2017). However, prior research has shown that machine learning systems often fail to correctly classify images after small perturbations to the original image. While these "adversarial attacks" (Chakraborty et al. 2018; Biggio et al. 2013), define a worst-case scenario for a given data pipeline, imperfect data is a natural result of any sufficiently complex system (Pearson 2005). In this work, we focus on intentional perturbations to the input space where the goal is to *evade* a classifier, but similar perturbations are a natural consequence of modern neural network architectures and hardware setups (see Sect. 2.1). Prior research has shown that proper data sanitation, anomaly detection, and model retraining are effective ways to combat adversarial attacks (Chakraborty et al. 2018; Grigorescu et al. 2020; Li et al. 2016; Wang et al. 2019). However, even state of the art defenses decrease the accuracy when compared to the un-defended (control) model, suggesting that the actual ability to generalize beyond laboratory test cases has been overestimated in the literature. This has been noted before (Biggio et al. 2013; Carlini et al. 2017; Wang et al. 2019; Li et al. 2016) and we confirm it below (see Sect. 7). Furthermore, recent research (Croce and Hein 2020) has shown that most defenses have worse performance against adversaries not tested at the time of publishing, arising from the tendency to only publish the 'best' results or better methods and hardware *that become* available to subsequent researchers.

Further questions about the feasibility of truly 'safe' artificial intelligence (AI) have been raised. For instance, it has been proven that no matter where we draw our boundary conditions, there exists an attack that will confuse any (non-perfect) discriminator or shift its boundary conditions (Dohmatob 2019). Additionally, while attacks are always possible on paper, a cost-aware analysis can reveal the feasibility of such attacks in practice. It is necessary for any safety-critical system to be robust to these attacks because, as we demonstrate, many classes of attacks are reliable even when we restrict perturbations to a single byte (see Fig. 11).

While we are unable to demonstrate safety-critical computer vision models, there is some remaining optimism due to techniques like network pruning (Sehwag and Wang 2019; Cosentino 2019; Jian et al. 2022), regularization (Ross and Doshi-Velez 2018; Jakubovitz and Giryes 2018), genetically evolved neural network architectures (Sinn et al. 2019), and FIRENETS (Colbrook 2021). However, the efficacy of these models is reported only with test-set accuracy numbers (see Eq. 1) that do not reflect the marginal computational cost of the techniques—which is consistently significant. In practical, real-time, and/or embedded systems, this is could make the technique unusable. Therefore, for a practical analysis, we need a metric that encodes both the change in accuracy and the computational cost of that change. Furthermore, the already-existing regulatory requirements for safety-critical electro-mechanical systems (see: Sect. 2.5 and Sect. 5) require such a high degree of precision that traditional error estimation techniques (e.g. traditional test-train split methodology) would be impractical to evaluate for every software change, despite that evaluation being a legal necessity (see Sect. 2.5 and Sect. 5). Furthermore, in order to estimate the confidence region, one must evaluate the techniques across the set of feasible hyperparameters–an often neglected practice in the literature which is frequently centered around marginal gains on benchmark data (Desislavov et al. 2021). This should also include any signal pre-processing techniques (see Sect. 4.4), any output post-processing techniques (see Sect. 4.4.6), and any attacks (see Sect. 6), as well as the traditional model hyperparameter optimization.

As such, we evaluated a large suite of proposed attacks and defenses in the contexts of accuracy, worst-case failure rate, and computation time. We show that every model defense configuration reduces the *accuracy* on benign (unperturbed) data. We show that, even when a particular defense decreases the *failure rate* against a given attack, that that behavior is inconsistent across distance measures and attack types. Most importantly, by using adversarial attacks to estimate the upper bound of the failure rate (see Section 14), we conclude that each and every tested configuration fails to meet safety-critical standards by a wide margin.

1.1 Contributions

- We show that even state-of-the-art defenses fail to make models that meet safety critical standards even if they tend to marginally improve the failure rate.
- We apply time and cost analysis for both the attacks and the defenses, something rarely done in the literature.
- We provide new insight into the robustness versus accuracy problem.
- We establish a standards-based framework for testing safety-critical computer vision systems in a way that meets regulatory standards without needing an infeasible number of test images.
- We survey a large suite of attacks and defenses to examine how each defense fares against each attack, measuring accuracy, worst-case failure rate, and run-time requirements in the context of safety-critical systems.

2 Background

Machine learning, artificial intelligence, and automated data collection are increasingly used in safety-critical applications like autonomous vehicles (Grigorescu et al. 2020; Al-Qizwini et al. 2017), medical imaging (Ching et al. 2017; Sahiner et al. 2019), and industrial control (Fukuda and Shibata 1992; Monmasson et al. 2011). Convolutional neural networks (CNNs) have demonstrated unparalleled accuracy in image classification tasks; however, CNNs have been shown to be very fragile to adversarial attacks (Miller et al. 2020; Finlayson et al. 2018). Research points to societal trust in fully automated banking thanks to, among other things, verifiable transactions and guarantees from the issuing institution (Srinivas Acharyulu and Seetharamaiah 2015). However, when it comes to real-time, safety-critical deep learning, models are rarely reproducible or verifiable (Tsipras et al. 2019; Carlini et al. 2017). Despite the drawbacks of deep-learning, modern aviation relies on an array of sensors to make largely automated decisions, relying on a framework of component testing and simulation (Rierson 2017). While similar test suites for adversarial robustness have been proposed (Carlini et al. 2017; Chakraborty et al. 2018), they rely on *de facto* accuracy goals rather than a solid theoretical and legal framework. The problems with this are plentiful. Historically, marginal gains have relied on exponentially larger models to produce increasingly marginal gains (Desislavov et al. 2021). These models rely on increasingly larger datasets (Desislavov et al. 2021; Vapnik et al. 1994; Anselm et al. 1989), which increasingly come from fewer sources (Koch et al. 2021), leading to gender-biased models (Lu et al. 2020), racism (Buolamwini et al. 2018), and fatal design errors (Banks et al. 2018) This is a trend that goes back decades (Corsaro William 1982; Ramirez and McDevitt 2000; Buolamwini et al. 2018), leading to, for example, significantly higher fatality in car accidents for female-bodied people (Evans and Gerrish 2001) or neural networks that unintentionally encode racial information from medical imaging data alone (Gichoya et al. 2022). Furthermore, data collection can be expensive (Roh et al. 2019), raises serious privacy concerns (Bloom et al 2017), increases time to market (Lam 2004), and impedes development speed (Zirger and Hartley 1996). Furthermore, research is focused on metrics that tend to be optimistic at best (Madry and Makelov 2017).

2.1 Image classification systems

In general, an image classification system, K, attempts to parse some image input signal, x, and output one of k class labels, $\hat{y} = K(x)$, with $\hat{y} \in \{1, \dots, k\}$. Each image is represented as a multi-dimensional array of $n \times m$ pixels, with bit depth b and color depth c, such that they are of size $\frac{m \cdot n \cdot b \cdot c}{8}$ bytes. When the model is a neural network, the images are passed into a composition of 'layers', each layer typically performing an affine transformation followed by a non-linear element-wise transformation (called an activation function). The free parameters of such a composition of layers (called an 'architecture') are found by minimizing a loss function, $L(y, K(x))$, that penalises differences between a true label, y, and an estimate, $\hat{y}$. When the problem is a multi-class classification, the loss function could, for instance, be the categorical cross-entropy loss (Biggio et al. 2013; Tsipras et al. 2019; Croce and Hein 2020; Carlini et al. 2017).

Adversarial vs. Benign

The accuracy is measured as either *benign* or *adversarial accuracy*. The former refers to the model performance on the original dataset (denoted unperturbed/benign/ben.) and the latter refers to a dataset generated by an attacker crafted to fool the model (denoted perturbed/accelerated/adversarial/adv.). In general, electro-industrial safety systems are governed by the International Electrotechnical Commission, IEC 61508 (International Electrotechnical Commission 2010), and medical software in particular requires continuous failure rate testing adding a massive computational burden to the development phase as governed by IEC 62034 (International Electrotechnical Commission 2006). In general, acceptable risk is expressed as a matrix (see Table 1) where these classes are known in the standard as the safety integrity level (SIL), which then corresponds to different failure modes for components that act on-demand (e.g., medical imaging) or ones that act continuously (e.g., object detection in autonomous cars). In general, for safety-critical systems, we aim for SIL levels III or IV, corresponding to failures that lead to injury or death respectively. Additionally, SIL levels I or II are generally considered to be unacceptable. In the context of safety-critical systems, whether the component be hardware or software, each component must meet the requirement in isolation, raising questions of legal compliance for any system that relies on proxy models, attack detection, or any other type of out-of-band component to ensure safety.

2.2 Adversaries

In general, an *attacker* seeks to maximize the loss against a given model rather than to minimize it. This is accomplished by perturbing samples from one class so that they fall

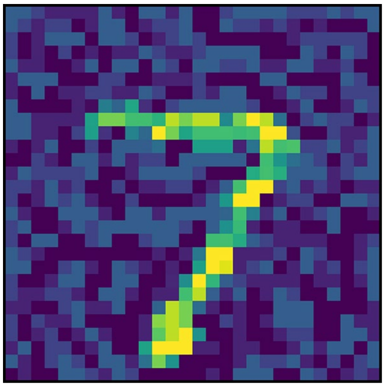

Fig. 1 A ‘7’ perturbed with adversarial noise such that the model perceives it to be a ‘2.’

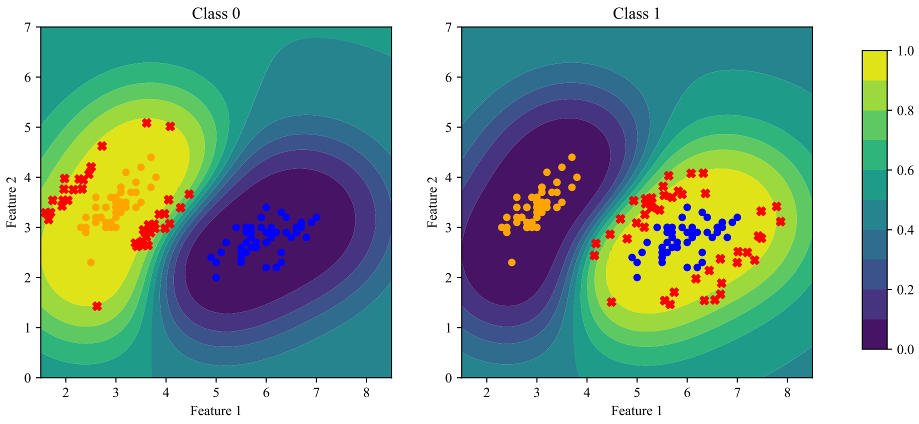

Fig. 2 Orange (class 0) points, blue (class 1) points, and red (adversarial) points in a contour map for a radial basis function support vector classifier. The contours reflect the confidence levels for a given sample and class. The bright yellow regions indicate areas of strong positive confidence and the purple areas indicate strong negative classification, and grey represents an uncertain classification. As we can see, it is rather trivial to shift the classification of a given sample towards the ambiguous regions (teal colored)

within the highly confident region of another, incorrect class. That is, attacks, by definition, are the worst-case perturbations of a given sample for a given model. While the literature focuses mainly on intentional adversaries, we posit that small perturbations of the input space are inevitable given the nature of real-world systems and that adversarial attacks simulate these failures. That is, things like calculation error, lens flare, lens aberration, dust, sensor failure, low-light conditions, and precipitation will all create noise that could inadvertently become adversarial. So, in an effort to measure and minimize these failures, we evaluate models against several possible attacks that attempt to *induce* different types of failures (see Sect. 3), each of which is subject to its own optimization criteria.

Figure 2 depicts a radial basis function support vector machine, classifying the points into orange (class 0) or blue (class 1). However, the red points indicate successfully generated adversarial examples. Figure 1 shows an example of a ‘7’ that has added adversarial noise such that the classifier sees it to be a ‘9’. In related surveys (Dohmatob 2019; Biggio et al. 2013; Bect et al. 2017; Chakraborty et al. 2018; Carlini et al. 2017; Croce and Hein 2020), a model’s adversarial robustness is defined as its performance accuracy against a given adversary. A thorough examination of such attacks are explored in Sect. 3 below, but, in general, an attacker perturbs an image such that the perturbation distance, d, is less than or equal to a threshold value, ϵ, specified by the experimenter, under some conception of distance. The evaluations in this study examine both the ℓ_∞ and ℓ_2 norms for the gradient-based attacks and ℓ_1 in other contexts, which consider perturbations no larger than ϵ, where ℓ_∞ and ℓ_2 allow for perturbations across the entire feature space, and ℓ_0 restricts the number of perturbed features. In our study, we restrict this perturbation distance to be 1 byte, as is typical in the literature (Madry and Makelov 2017; Miller et al. 2020; Biggio et al. 2009, 2013).

Attack Strength Since adversarial perturbations, by definition, are the perturbations that maximizes the model loss through various methods, each approximates a different worst-case scenario. The 'strength' of an attack is generally related to the magnitude of these perturbations (Carlini et al. 2017), and is measured by retrospective evaluations of model accuracy in which a 'strong' attack induces more loss. It is necessary to evaluate against the strongest possible attack for a given model and data set, but the strength of an attack is always contextual, since the magnitude of a perturbation must be measured with respect to some normed vector space, is specified in advance, and subject to real-world constraints. Additionally, we know that models optimized to prevent one attack do not necessarily generalize to other attacks (Carlini et al. 2017), especially across distance metrics.

2.3 Defenses

The attacks outlined above are capable of breaking state of the art image recognition models. However a variety of defenses have been proposed that act on the dataset or the model output. Those broadly fall into categories that seek to identify an attacker and isolate them from the model API, ones that seek to isolate tainted examples from the database, or ones that attempt to mitigate all potential attacks during run-time. Below, we outline a wide variety of defenses proposed over the last several years and measure their effect on the failure rate of various models. We have excluded model transformation defenses and secondary detection models since those sidestep the problem of making a given model more robust and do nothing for the IEC requirement that each electro-mechanical component meets the regulatory standard in isolation from each other component (see Sect. 5).

2.4 Attacker's knowledge

While the general assumption is that an attacker has whitebox access to an entire pipeline (including data, model weights, and output), that does not necessarily need to be the case (Chen et al. 2020; Chakraborty et al. 2018). While some attacks do need whitebox access, prior research (Chen et al. 2020; Blaine 2010) has shown that a surrogate model and data-set can be used to approximate K using a proxy model $\widehat{K}$, built using the class labels provided by the model at test-time, that is sufficient for creating strong adversarial examples. Tramèr et al. Tramèr et al. (2016) examined popular machine learning as a service platforms that return confidence values as well as class labels, showing that an attacker can build a proxy model by querying $p + 1$ random p–dimensional inputs for unknown $p + 1$ parameters. Further research (Fredrikson et al. 2015) was able to reverse engineer the training data-set through black-box attacks against a model that returns confidence levels, with the caveat that the inferred data might be a meta-prototypical example that does not appear in the original dataset.

Table 1 Acceptable Failure Rates for different SIL levels in which a single death is possible, measured in failures per second

SIL	On-demand operation	Continuous operation
I	$[10^{-6}, 10^{-5})$	$[10^{-10}, 10^{-9})$
II	$[10^{-7}, 10^{-6})$	$[10^{-11}, 10^{-10})$
III	$[10^{-8}, 10^{-7})$	$[10^{-12}, 10^{-11})$
IV	$[10^{-9}, 10^{-8})$	$[10^{-13}, 10^{-12})$

Fortunately for our attacker, such examples are still useful for determining the underlying data distribution, even if they manage to preserve some of the privacy of the original dataset. Shokri et al. Shokri et al. (2017) presented a membership inference attack that determines whether a given data point belongs to the same distribution as the original training data using a set of proxy models. Below, we examine the efficacy of several attacks from the perspective of loss as well as the functional query bandwidth.

2.5 Metrics

The ISO standards (International Standards Organization 2018) define the Safety Inegrity Level (SIL) in failures/per hour, which we have converted to failures per second in Table 1. If assume that *accidental* adversarial errors are possible in real-world systems due to things like dust, lens flare, component failure, packet loss, etc., it naturally follows that the adversarial failure rate is an estimate of the models behavior at the edge or in the 'worst-case scenario'. That is, the *adversarial failure rate* is an estimate of the upper bound of the real-world failure rate in adverse but otherwise mundane circumstances.

(a) Accuracy: The accuracy is defined as

$$\text{Accuracy} = 1 - \frac{\text{False Classifications}}{\text{Total}} = 1 - \eta, \tag{1}$$

where *Total* is the number of tested samples, *False Classifications* refers to the number of objects that were incorrectly categorized by a given model [, and η is the generalized error rate. In practice, η is generally assumed to be the accuracy on a 'test set', with samples from a distribution assumed to be identical to the training set. Elsewhere, we refer to this 'test set' accuracy as the 'benign' accuracy or with the subscript 'ben.' such that the test accuracy is $\eta_{ben.}$. In addition to this metric, we include metrics for a variety of signal processing techniques (see: Sect. 4) where the unaltered signal is designated as 'control'. Finally, we include many sets of test sets specifically crafted to be 'adversarial' (see: Sect. 3), which are denoted with the subscript 'adv.'.

However, due to the large number of samples required by regulatory standards and the strenuous testing requirements of safety-critical software (see Sect. 5), these evaluations become an infeasible way to verify that a model only fails once across the required number of samples (see Table 1), especially if we would like to be highly confident of that estimation.

Failure rate

Instead of evaluating every software change in our pipeline against the legally required $[10^7, 10^{12})$ number of samples, we can measure the precise failure rate (elsewhere λ) with a much smaller number of samples if we measure it with

$$\text{Failure Rate} = \frac{\text{False Classifications}}{\text{Total Time (s)}} = \lambda, \tag{2}$$

where *False Classifications* is the number of misclassified samples, and *Total time* refers to the total time it takes to classify all the samples.

2.6 Robustness

Robustness, then, is a measure of how well a model resists these *induced* failures. In this survey, we examine how several different model and data transformations (see Sect. 4) influence this property for a given model architecture and dataset. We measure the efficacy of a given model change, using the *Percent Change in Accuracy* ($\%\Delta ACC$):

$$\%\Delta\text{ACC} = \frac{\text{Acc.} - \text{Control Acc}}{\text{Control Acc}} \cdot 100 \tag{3}$$

where *Acc* refers to the *accuracy* as defined in Eq. 1 and *Control* refers to the performance of the unchanged model on the benign (*Ben.*) dataset. This measures the marginal risk of failure for a particular model change (defense) in the adversarial case when compared to the benign case. We also defined the the metric *Relative Change in Failure Rate* ($\Delta\lambda$):

$$\Delta\lambda = \frac{\lambda_{\text{control}} - \lambda}{\lambda} \tag{4}$$

where λ refers to the failure rate, *Control* refers to the unchanged model. Taken together, these two metrics allow us to measure the marginal risk of a given defense in both the benign and adversarial circumstances. In both cases, a positive number indicates an improvement in relative risk and a negative number indicates a worsening of relative risk, Eq. 3 in the context of accuracy and Eq. 4 in the context of failure rate.

2.7 Hyperparameter selection

For many attacks, hyper-parameters such as the targeted false confidence threshold, step size, batch size, number of iterations, and distortion norm must be specified in advance by the attacker (Carlini et al. 2017). Furthermore, because many of these are drawn from a continuous (and therefore infinite) space, finding a strong attack is computationally expensive and finding the strongest possible attack is at least NP-Hard (Carlini et al. 2017), with the problem exacerbated by the extreme non-linearity of CNNs. Even more concerning, there is not yet a mathematical foundation for what constitutes a 'good' attack, relying only on after-the-fact evaluations of model accuracy. By examining a large hyper-parameter space, we demonstrate how each defense generalizes across the feasible attack spectrum rather than relying on a single canonical evaluation metric.

2.8 Attack and defense cost analysis

Like in cryptography, the fundamental limit for an adversary has to do with computational cost (Hoffstein et al. 2010). For example, model inversion attacks become pointless if it is computationally more expensive to steal a model than it is to train one. Likewise, model defenses are only as useful insofar as they have the ability to be deployed in existing real-time systems. As such, we examine the cost of various defenses as well as the number of queries and query rate of various attacks. The best modern methods are limited to computationally expensive techniques like reject on negative impact (Blaine 2010), Bayesian subset analysis (Bect et al. 2017), and model post-processing techniques that degrade accuracy with added computational cost (Athalye et al. 2018; Tramèr et al. 2017; Li et al. 2016), making them unsuited for the task of improving our ability to reliably generalize. For most attacks, we tested the perfect knowledge scenario, but we have also included the 'HopSkipJump' attack (Chen et al. 2020) to model the worst case for an attacker that only has access to a standard machine learning application programming interfaces (APIs) which only exposes hard class labels.

3 Attacks

The following section outlines a variety of attacks, broken into three categories: gradient-based attacks, gradient approximating attacks, and universal attacks. For the sake of the reader, a collection of generated adversarial samples follows the mathematical descriptions in the subsection following the aforementioned trio (see: Sec. 3.4).

3.1 Gradient-based attacks

The seminal work on adversarial attacks in the context of modern neural networks was written by Madry et al. Madry and Makelov (2017) and an important follow up work was written by Carlini and Wagner Carlini et al. (2017). Both operate under the condition that the attacked model has a gradient that is known to the attacker. Each of these can be considered white-box attacks because they are given access to the model output and the model gradient.

3.1.1 FGM

The fast gradient sign method (FGM) (Goodfellow et al. 2014) is the most basic and fastest such attack. It is defined by the step,

$$\widetilde{x} = x + \epsilon \cdot \operatorname{sgn}\Big(\nabla_x L\big(y, K(x)\big)\Big),$$

where L is the loss function, as described above, $\nabla_x L$ is the gradient of L with respect to the input x, the ϵ is the maximum perturbation distance, sgn is the element-wise signum function, and $\widetilde{x}$ denotes a generated adversarial sample. It is called 'fast' because it does not check the feasibility of the perturbation (if it is within a maximum distance, ϵ, as is done in other methods, see e.g. PGD below).

As such, it may not be as successful as other methods, but operates very quickly. We tested several different step sizes and norms for this method, enumerated in Figs. 11 and 18.

3.1.2 PGD

Projected gradient descent (PGD) (Carlini et al. 2017) takes a gradient step to increase the loss, but also includes a projection step, Proj_ϵ, that enforces the constraint of a perturbation distance q of at most ϵ with respect to some norm by projecting onto the feasible set (defined by the perturbation distance, ϵ). The iteration scheme of PGD is

$$x^{(s+1)} = \mathrm{Proj}_{\epsilon}\left(x^{(s)} + r \cdot \nabla_{x^{(s)}} L\big(y, K(x^{(s)})\big)\right),$$

where r is a step size, s is a sequence index, and $x^{(S)}$ denotes a generated adversarial sample after S steps. This iteration is repeated until a specified number of iterations, S, have been reached. The number of iterations is specified by the attacker, which ultimately determines the processing-time against a given model, scaling linearly with the number of iterations. We tested several different step sizes and norms for this method, enumerated in Figs. 11 and 18.

3.1.3 Carlini–wagner

Carlini and Wagner (CW) (Carlini et al. 2017) devised an attack that minimizes the perturbation distance subject to a distance constraint while maximizing the false confidence. The iteration scheme is for some constant, C,

$$x^{(s+1)} = \arg\min_{x} \|x^{(s)} - x\|_2^2 + C \max\left(\max_{j \neq t} g_j(x) - g_t(x) + C, 0\right)$$

where the g_j, g_t are discriminant functions and the goal is to minimize the perturbance with a penalty for not changing it to the target class. It also generalizes beyond the squared ℓ_2 norm. This method attempts to enforce attack quality by penalizing examples with low confidence and continuing to iterate on them until either a maximum number of iterations or the specified false confidence is reached. For our tests, we used the ℓ_∞ norm and a confidence threshold of 99%.

3.2 Gradient-approximating attacks

Further work sought to find attacks that did not rely on explicit gradient information. The *Deepfool* attack (Moosavi-Dezfooli et al. 2016) uses a quadratic approximation of the boundary condition to generate a separating hyperplane, while *Few-Pixel* and *Threshold* (Kotyan et al. 2019) use a search algorithm known as differential evolution rather than relying on explicit gradient information. Each of these attacks can be considered *grey-box* attacks because they rely on un-categorized model output but use that output to approximate the gradient, rather than rely on explicit access to the model weights.

3.2.1 DeepFool

The DeepFool attack (Moosavi–Dezfooli et al. 2016) seeks to find the smallest perturbation that causes a misclassification using a first-order Taylor approximation of the classifier. In essence, it seeks to find the minimal separating hyperplane between the target sample's true class and another. It assumes that it has access to the entirety of the model, including the confidence level (expressed as a logit) of all labels, the true label of a sample (x) and the model gradient. The iteration scheme is

$$x^{(s+1)} = \arg\min_{x} \|x^{(s)} - x\|_2$$

subject to

$$f(x^{(s)}) + \nabla_{x^{(s)}} f(x^{(s)})^T (x^{(s)} - x) = 0,$$

where $\|\cdot\|_2$ is the ℓ_2 norm, and where f outputs the logits of K, such that $K(x) = \phi(f(x))$, with output activation, ϕ. Run-time is determined largely by the number of iterations, S, and the number of gradients *w.r.t.* $L(x)$ used to estimate $f(x)$, which we set to 10 such that the gradient was calculated for each class. The existence of any such attacks with a distance than or equal to some some specified robustness threshold ($\epsilon \leq \epsilon_{critical}$) should immediately cause concern.

3.2.2 Few-pixel

The few-pixel attack (Kotyan et al. 2019) (Pixel) attempts to maximize the loss by iteratively finding the least robust pixel set and perturbing it by less than some specified threshold, ϵ. That is, this attack seeks to maximize false confidence while minimizing the number of perturbed pixels. The iteration scheme is

$$x^{(s+1)} = \arg\max_r L\big(y, K(x^{(s)} + r)\big) \text{ subject to } \|r\|_0 \leq \epsilon,$$

where r is the perturbation and ϵ is the perturbation distance in pixels, typically of just one pixel. This attack uses differential evolution to simultaneously optimize for both loss and the number of perturbations. In our case, we limited this to a single perturbation, but did not restrict epsilon beyond the normal [0, 255] range. Since perturbation distance for a single pixel is not restricted, we restricted the number of distorted pixels to one of these: [1, 2, 4, 8, 16]. Unlike the aforementioned gradient-based attacks, this method generalizes to classification systems that lack gradients (e.g., decision trees). This differs from the the *Threshold* and *Adversarial Patch* attack below by optimizing for the fewest number of changed pixels (ℓ_0 norm) rather than the perturbation distance (ℓ_2 or ℓ_∞).

3.2.3 Threshold

The threshold attack (Kotyan et al. 2019) is similar to the few-pixel attack (in that it uses differential evolution as the optimization algorithm), but uses the ℓ_∞ norm rather than the ℓ_0 distance. This method, like the Carlini Wagner (CW) method, attempts to generate examples that are both false and highly confident. However, it uses a complicated algorithm (differential evolution) rather than simple, linearized methods (as in CW). The iteration scheme is

$$x^{(s+1)} = \arg\max_r L\big(y, K(x^{(s)} + r)\big) \text{ subject to } \|r\|_\infty \leq \epsilon$$

where ϵ is the targeted perturbation threshold of.03. Unlike the aforementioned gradient-based attacks, this method generalizes to classification systems that lack gradients (e.g., decision trees). This differs from the the *Few-Pixel* and *Adversarial Patch* attack below by optimizing for largest change in loss (ℓ_1 norm) rather than fewest number of pixels (ℓ_0 norm) or the perturbation distance (ℓ_2 or ℓ_∞).

3.3 Universal attacks

Even further work has sought to go beyond computationally-intensive approximations for each attacked sample or model query. The *Adversarial Patch technique* (Brown et al. 2017) uses the aforementioned differential evolution algorithm to generate an additive noise sample that maximizes the loss for all samples and is considered a *grey-box* attack because it relies on un-categorized model outputs but not explicit access to the model weights. However, we include it here because the nominal image patches generated by this technique are meant to generalize to unseen data as the number of API queries increases, meaning that this can be trained using data wholly disconnected from the model-builder's dataset. The *HopSkipJump* attack (Chen et al. 2020) has been shown to minimize the number of API queries required to break any model. As such, it is considered a *black-box* attack.

3.3.1 Adversarial patch

The Adversarial Patch attack (Brown et al. 2017) (Patch) uses the same differential evolution algorithm as the Threshold Attack. However, instead of optimizing for a threshold or confidence level, it seeks to change each class into any other by applying an image patch that is not unique to a given image, but is universal for a given dataset. This single generated image patch is added to each image and is intended to cause a generic misclassification, regardless of the original class. We know that these attacks are quite general for a given dataset and not specific to a given model (Brown et al. 2017; Xiao et al. 2021), raising serious concerns about an attacker's ability to generate universal and offline attacks for a given set of data. Like with the Pixel attack, the per-pixel distortion is not restricted, so we restricted perturbation to a percentage of the benign image, using the hyper-parameters [.03, .1, .25, .5, 1.0]. Unlike the aforementioned gradient-based attacks, this method generalizes to classification systems that lack gradients (e.g., decision trees). This differs from the the *Threshold* and *Adversarial Patch* attack below by optimizing a single perturbation that works for every image rather than minimizing the geometric perturbation distance (ℓ_2 or ℓ_∞ norm) while minimizing the number of pixels required for said universal patch (ℓ_0).

3.3.2 HopSkipJump

The HopSkipJump attack (Chen et al. 2020) (HSJ) is a query-cost-aware model that acts on hard class labels rather than confidence levels (i.e., a blackbox attack). It finds a point on the boundary (using binary search between the initial attacked point and a point on the other side of the decision boundary), and approximates the gradient at the boundary using Monte Carlo sampling in an offline manner. Then, a step is taken in the approximated gradient direction, the model is queried again with the new points, and the process is repeated. The procedure iterates the step,

$$x^{(s+1)} = \mathrm{Proj}_{\epsilon,x^{(0)}}\left(x^{(s)} + r^{(s)}\frac{\widetilde{\nabla}_{x^{(s)}}Q(x^{(s)})}{\|\widetilde{\nabla}_{x^{(s)}}Q(x^{(s)})\|_2}\right),$$

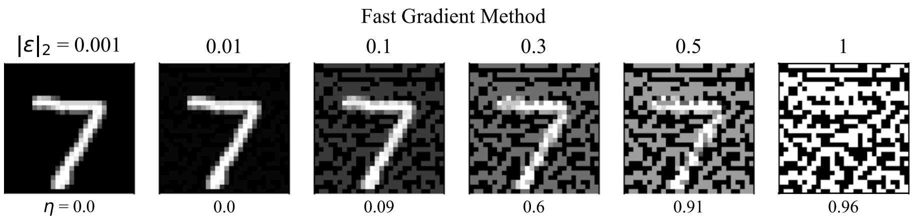

Fig. 3 The Fast Gradient Method (FGM) doesn't enforce any kind of feasibility criteria, resulting in salt-and-pepper noise as the ℓ_2 approaches the standard deviation of the normally distributed dataset

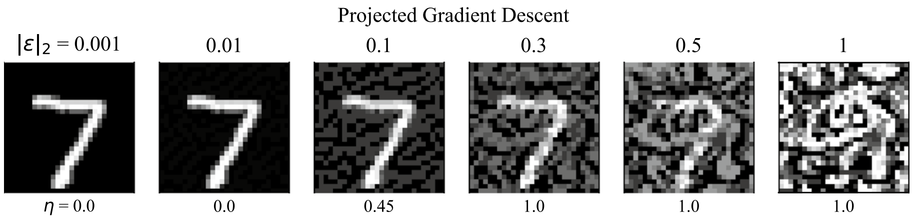

Fig. 4 The Projected Gradient Descent Attack (PGD) projects the adversarial example back onto a sphere of a fixed radius, yielding noise that more closely approximates handwritten digits

where $x^{(s)}$ is an adversarial sample, s a sequence index, $\text{Proj}_{\epsilon,x^{(0)}}$ is an ℓ_2 projection of a point onto a sphere of radius ϵ, centered at the initial point $x^{(0)}$, and $\widetilde{\nabla}_x Q(x)$ is the Monte Carlo estimate of the gradient of Q at x, and

$$Q(x) = \max_{c \neq c^*} K_c(x) - K_{c^*}(x),$$

where K_c is the model output for class c, and c^* is the class of the initial point, $x^{(0)}$. This attack's run-time is controlled by the number of (offline) gradient estimations, the number of random samples per (online) query, and the number of iterations of both parts of the algorithm. This attack finds an adversarial example that is close ($\epsilon \leq \epsilon_{critical}$) to the original sample, but still causes a misclassification. This model is query efficient as it uses a small number of API queries to approximate the gradient and then uses this information to approximate the gradient near the class boundary and optimize the perturbation distance of the attacked sample. Also, because this attack approximates the gradient rather than relying on explicit whitebox access to it, it can be used to attack non-differentiable models (e.g. random forests). That is, this is one of the most universal black box attacks.

3.4 Attack samples

To aid the reader, we have visualized a single attack sample for each attack on the undefended (control) model, depicted in Figs. 3, 4, 5, 6, 7, 8, 9, 10. Additionally, we have provided the failure rate, η, defined in Eq. 2, and tested on 100 samples from the MNIST dataset. For each attack, we also vary a distance parameter, determined by the optimization criteria for each attack outlined above. For the sake of clarity different distance metrics are denoted with a distance of *epsilon* subject to some norm, $| \cdot |_n$ such that ℓ_2 norm of ϵ is $|\epsilon|_2$

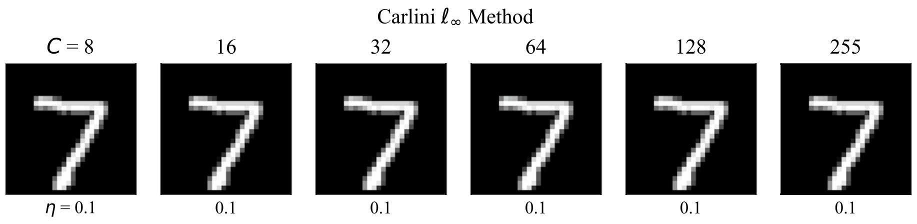

Fig. 5 The Carlini-Wagner ℓ_∞ method includes an added confidence constraint that only returns an adversarial example $\Longleftrightarrow$ $L(x^{s+1}) - L(x) \geq C$. Depicted here is the unperturbed 7 because this method was unable to find an adversarial example for these constraints. Despite this, we can see that for 10% of the samples are consistently misclassified with a high degree of false confidence

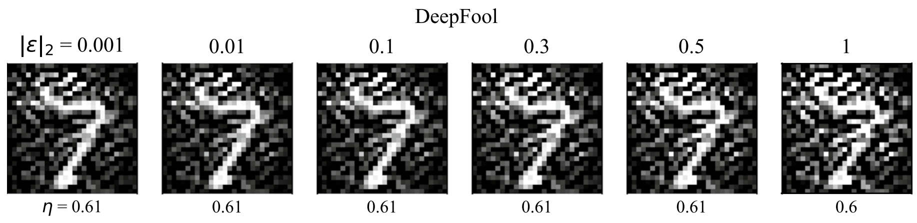

Fig. 6 The DeepFool Method, rather than constraining perturbations to an ϵ-sphere around the sample, finds perturbations that extend beyond the approximated boundary by a distance of at least ϵ. Regardless of the distance constraint, DeepFool is very effective

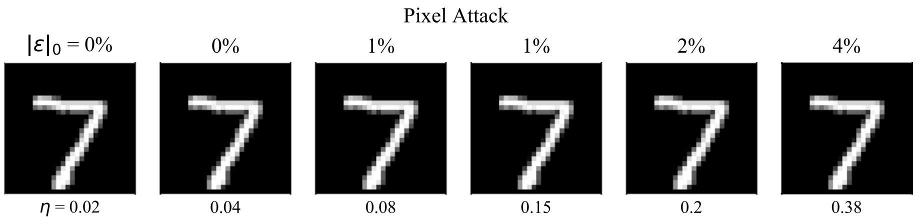

Fig. 7 The Pixel Attack (Pixel) seeks to maximize the loss while minimizing the number of perturbed pixels. Here we see that even a small perturbation in original image can lead nevertheless consistently induce failures. Here, ϵ_0 denotes the ratio of perturbed pixels to the total

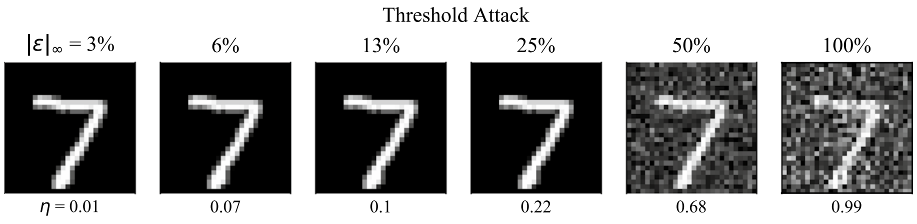

Fig. 8 Rather than minimizing the number of pixels, the Threshold Attack (Thresh) optimizes for the smallest perturbation possible. Here, ϵ_∞ denotes the ratio of the applied to noise to the maximum possible value (255)

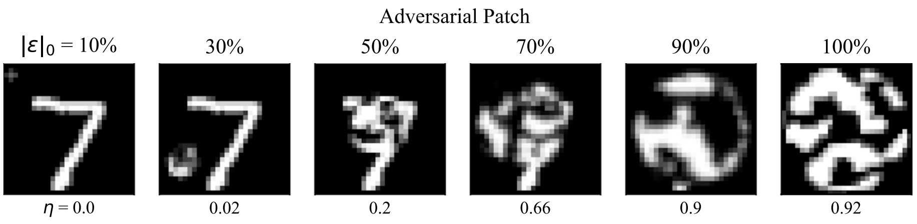

Fig. 9 This is one of the two most dangerous attacks as it can consistently find image patches (the noisy circle depicted above) from only a small number of samples (n = 100) that can often fool the classifier, regardless of original image or class. Obviously, as we replace all of the pixels with adversarial noise, the classifier becomes mostly useless, but this attack is still concerningly effective when only a fraction of the image has the adversarial noise. Here, ϵ_0 denotes the ratio of perturbed pixels to the total

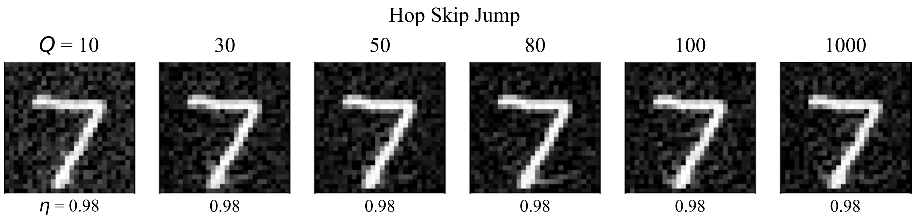

Fig. 10 The Hop Skip Jump Attack (HSJ) uses a second-order approximation of the classification boundary in an offline manner to find adversarial examples that first maximize loss then minimize for perturbation distance. This attack is uniquely concerning because it 1) works on any model and 2) is surprisingly effective even when our attacker is restricted to a small number of queries per batch (here, denoted as Q)

4 Defenses

4.1 Attacker identification

We must assume that at least some of the user inputs will 'adversarial', even if that adversary is sensor failure and not an intentional attack. Identifying and isolating this adverse input may not require a perfect anomaly detection system, but could draw from graph theoretical representations to identify and isolate networks of distributed attackers, allowing the model API provider to revoke access or otherwise isolate the attack effects from the models. While this has been done in the context of social networks (Daya et al. 2019), these techniques can easily be fooled with intermittent attacks (Park et al. 2018), distributed attacks (Aljuhani 2021), or something as simple as a quadratic approximation of the model (Chen et al. 2020). For web services more generally, legitimate users are often identified by using CAPTCHA (von Ahn et al. 2003), but that it not a solution for an API meant to be accessed by software. Furthermore, outsourcing this to a secondary component would run afoul of the IEC requirement that each component meet regulatory standards in isolation from all other components (see Sect. 5). However, even if we assume all samples are generated by legitimate users with guaranteed data integrity, we still cannot be confident that 'adversarial' noise will not be generated inadvertently by routine phenomena like sensor failure, dust, low-light conditions, lens aberration, precipitation, or another mundane cause. One possible approach is to eliminate 'bad' samples at run-time.

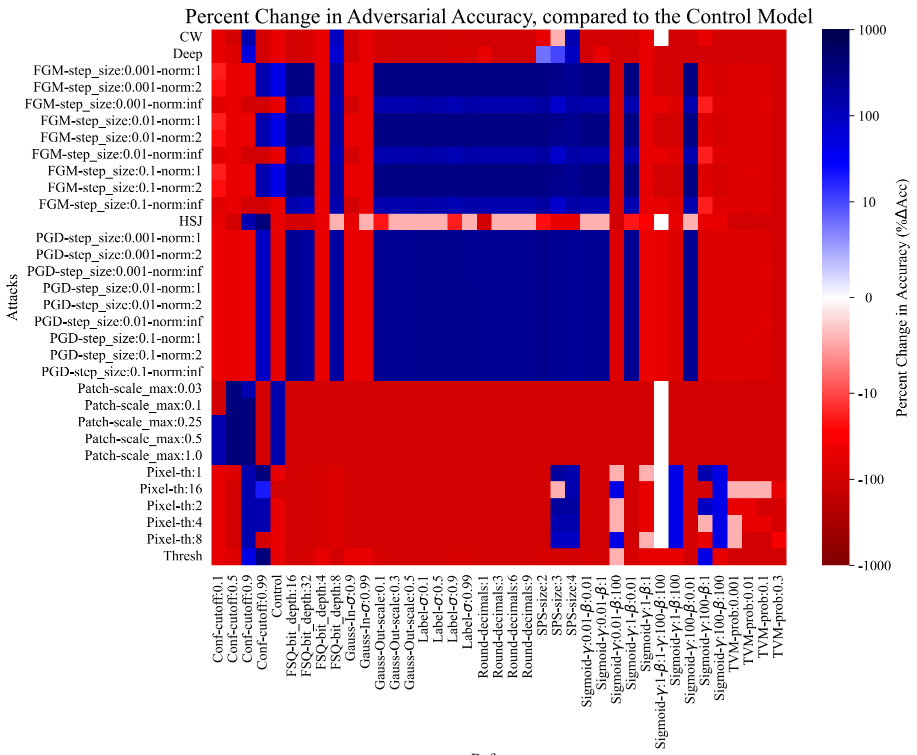

Fig. 11 The percent change in adversarial accuracy of each attack against each defense for CIFAR-10. As we can plainly see, no defense was able to improve the failure rate across all tested attacks. Red indicates that defense made a model worse. Blue indicates an improvement. White indicates no change

4.2 Subset analysis

Subset analysis (Paudice et al. 2018) examines how a particular sample changes the model's performance (Paudice et al. 2018). By exhaustively comparing the model accuracy on various subsets of data, it attempts to isolate adversarial samples by removing ones that lead to worse-performing models (i.e. the sample is 'bad'). If the database is large, this becomes an incredibly expensive task. This method also assumes that all 'bad' data is, in fact, adversarial and not a legitimate measurement of real world circumstances. Another method, called 'Subset Scanning' (Cintas et al. 2020), examines the hidden layers of a neural network to ensure that a particular sample looks 'typical' as it passes through the models layers rather than just at the final layer. This comes with the added cost of tracking the model through each of these layers for each of these samples, which becomes infeasible in real-time systems due to the size and complexity of neural networks and their associated datasets.

4.3 Attack mitigation

Rather than relying on a generic framework for detecting and preventing all attacks, as discussed above, there are mathematical foundations for avoiding the impacts of adversarial attacks during model creation. These are either 'pre-processing' defenses or 'post-processing' ones in which alter either the data (pre-processing) or the model output (post-processing) to mitigate the risk of an attack. In general, the goal of these defenses is to reduce the noise in the input, or to reduce the precision in the output, corresponding to the pre- and post- techniques. In this way, we seek to examine how modern neural architectures perform on the generalized 1-byte spherical perturbation that surrounds a true example of a given class, rather than rely on external components to identify and mitigate the attacker.

4.4 Pre-processing

There are a variety of ways to change the data before training so that the resultant model is more robust to adversarial perturbations.

4.4.1 Gaussian augmentation

The most straight-forward defense (Gauss-In) is where random Gaussian noise is added to the input data and the model trained without modifying the class labels (Zantedeschi et al. 2017). If we replace real samples with noisy ones, the processing time and space are marginal. We tested noise with standard deviations of .9, .99, and .999 on data that was zero-centered and normalized.

4.4.2 Label smoothing

The label smoothing (denoted Label) defense (Warde-Farley and Goodfellow 2017) sets a cap on the confidence level for a given model output. If the output layer outputs a number higher than this cap, it uniformly distributes the difference across all classes. In this way, it obscures the model output, reducing the effective query rate for the attacker. In our experiments, we set this threshold to be 99%, 99.9%, or 99.99% which itself is far below the regulatory standards outlined in Sect. 5.

4.4.3 Feature squeezing

The feature squeezing (FSQ) method (Xu et al. 2017) reduces the bit depth of the input image to a specified value, treated as tunable parameter, which hopefully increases the signal to noise ratio. The initial processing time merely requires setting some bits to zero which can be vectorized and parallelized, and scales with image size. However, the resulting model can use smaller data-types and potentially operate faster and require less memory. We tested bit-depths of 32 and 16 (the control is 64 bit and the images are 8 bit).

4.4.4 Total variance minimization

Total variation minimization (TVM) is an image de-noising techniques that dates back decades (Rudin et al. 1992). It exploits the fact that images with spurious details have high total variation. This defence is effective at preserving edges within an image, and encourages spatial homogeneity such that large jumps in intensity between neighboring pixels are penalized, leading to a smoother image, determined by some specified noise level. This minimization problem is non-trivial, and there are several specific and tailored algorithms for it Chambolle (2004); Hadj-Selem et al. (2018). Thus, we tested several combinations of the noise level (denoted ‘prob), enumerated in Figs. 11 and 18.

4.4.5 Adversarial re-training

Adversarial retraining (Retrain) is a method proposed by Croce et al. Tsipras et al. (2019), that appends adversarial examples to the training set, labels them ‘adversarial’ and trains a classifier on the new (larger) dataset. The first problem with this method is that the training time increases linearly with the number of re-training epochs, with twenty retraining cycles being recommended in the original paper. Furthermore, ‘adversarial re-training’ must be conducted against each type of attack individually since the topological characteristics of attacks vary widely. An extension seeks to encode ambiguity between an adversarial example and both the original and target class, called ‘confidence-calibration’ (Croce and Hein 2020) by changing the class label from an integer to a float that decays with distance from the ’true’ image. While it offers improved results over other types of adversarial re-training, it optimizes against a particular type of failure which inherently degrades performance against others (Carlini et al. 2017).

4.4.6 Post-processing

There are a variety of ways to change the the model outputs so that the user-exposed API reveals less information to the attacker.

Gaussian Noise This post-processing defense (Gauss-Out), like its pre-processing counterpart adds Gaussian noise, but in this case, it applies it to the model outputs (Lecuyer et al. 2019). Since it acts on a discrete output vector, the marginal cost is negligible in comparison to image processing. However, it’s efficacy is tied to reducing the accuracy of the API by an amount proportional to the variance of the added noise. That is, it reduces the number of useful output bits available to an attacker (as well as legitimate users).

High Confidence Thresholding Rather than decrease the precision of the output as in other techniques, this method (Confident) only returns model output if the confidence level exceeds some threshold specified by the model builder (Chen et al. 2021). While this does make it harder for an attacker to calculate a gradient step, the attacker can circumvent it by taking a step large enough to overcome the thresholding (i.e., by changing the model output by more than twice the cut-off value, ensuring that the classification ‘jumps over’ the obscured boundary) HopSkipJump is a query efficient model for doing exactly this (Chen et al. 2020), but any other method could be effective by simply increasing the perturbation threshold such that perturbation extends beyond the ‘fuzzy’ class boundary. In our experiments, we set this to be equal to 50%, casting the normal one vs. one problem into one vs. the sum of the rest, ensuring that the transformed model returns no classification if the confidence of the label does not exceed the total confidence of the other labels. To simulate scenarios in which low-confidence classifications are merely ignored (which, in practice, could mitigate an attacker as they generate their attack), we exclude all such samples from the accuracy calculations for this defence.

Rounded Instead of obscuring the output data with noise or only reporting highly confident answers, this method (Rounded) merely reduces the bit depth of all the reported confidence levels (Zhang and Liang 2018). Instead of a 64-bit output vector, this might be reduced to an 8-bit vector reducing the effective attack query rate by a factor of four. However, this harms the legitimate user by the same degree by reducing the precision of their queries as well. For our study, we used several different numbers of decimals, reflecting the number of base 10 digits in the set [0, 1] revealed by the API. enumerated in Figs. 11 and 18.

Reverse Sigmoid The Reverse Sigmoid (denoted Sigmoid) defense (Lee et al. 2018) changes the activation function from the rectified linear unit (ReLU) or the Weiserstrass σ -function to a function that retains the approximately linear behavior when the confidence is near 0 but instead of asymptotic convergence, model confidence eventually decreases rather than converging to one. The Reverse Sigmoid activation function, A, is defined as,

$$A(y_i) = \alpha^i(y_i - \beta(\sigma(\gamma\sigma^{-1}(y_i) - 1/2))),$$

where α, β, γ are scaling parameters, specified as hyperparameters, y_i is the class label of sample i, and σ is the logistic sigmoid function. Run-time requirements are basically identical to other activation functions as the goal of the model builder is to remain more-or-less along the linear section of the function. This has the effect of preserving $y = f(x)$ while ensuring that the class label, $\hat{y}, \neq K(x)$. In essence, this traps an attacker in a local minimum that is in the opposite gradient direction of the global minimum. Even if this defense is known to an attacker, the non-bijective nature of this function makes the model non-invertible (beyond specified thresholds) to gradient descent methods. Alpha is a scaling parameter that is determined by β and γ, both of which must be positive. Both β and γ were evaluated in a grid search of the set [.01, 1, 100] for each variable, yielding 9 combinations.

5 Safety critical computer vision: a framework for robustness guarantees

5.1 Safety critical computer vision

Since the bulk of the literature focuses on image classification systems (Dohmatob 2019; Biggio et al. 2013; Bect et al. 2017; Chakraborty et al. 2018; Carlini et al. 2017; Croce and Hein 2020), we chose to stress test them in the contexts of autonomous vehicles, medical imaging, and industrial control which are each governed by pre-existing standards across the different domains. Safety-critical software is already widely deployed in other electro-mechanical systems like vehicular braking systems (Chae et al. 2017), aviation (Rierson 2017), and medical implants (e.g., pacemakers) (Tuan et al. 2010). The International Standards Organization provides a safety-threshold of 10^{-9} failures per second for any life-threatening situation (International Standards Organization 2018) and 10^{-6} (International Standards Organization 2018) for any risk of harm, required of any automotive or aviation system governed by ISO 26262 (International Standards Organization 2018). For an autonomous vehicle trying to classify objects on the road, a false negative classification of, for example, a cyclist could lead to death; whereas, a false positive detection of a cyclist would be more likely to cause braking-related injuries that are less severe. For medical imaging, a false negative classification could mean loss of life; whereas, a false positive is less likely to cause grievous harm (but likely to be lead to expensive and unnecessary additional testing). Understandably, these correspond to the differing legal and technical requirements outlined in these international standards.

5.2 A framework for robustness guarantees

Since adversarial failure rate provides a worst-case estimate of the failure rate for a given context (see Sect. 2.5), we propose a safety critical testing framework that (1) evaluates the benign and adversarial failure rate across several attack metrics (as dicated by the safety-requirements of the system) (2) repeats those evaluations across a feasible hyper-parameter space to estimate a confidence interval for the values in (1), and then (3) rejects any model that does not consistently meet the standards outlined in Table 1 as unsuitable for safety-critical systems. The limitations of this approach are discussed below in Sect. 7.8.2.

Advantages Because of the relatively small run-time requirements of this approach (when compared to testing against massive in-distribution test sets), this method could, for example, act as a unit test in machine learning applications rather than relying on full-system integration tests to evaluate changes to a single model, signal processing technique, data storage format, or API access mechanism. It could also be used to highlight error-prone classes or other subsets of data to reduce error or create synthetic samples. Furthermore, by isolating changes and testing them as quickly as possible, it's much easier to parse cause and effect when compared to full-system integration tests that could include many changes from many different development teams and require live and potentially dangerous systems (like cars or MRI machines) to effectively test. To further increase development velocity, we propose metrics Equ. 3 and 4 as standards for evaluating not only the efficacy of a given change, but as tools to quantify the marginal risk associated with each change, as dictated by the ISO 26262 standard (International Standards Organization 2018).

6 Experimental methods

In this study, we evaluated the accuracy (see: Sect. 1) and the failure rate (see: Equ. 2) for a variety of attacks (see: Sect. 3) and defenses (see: Sect. 4) using the methods discussed in detail in the previous section. In our experiments, we go beyond the in-distribution train/test split typical in machine learning research (Equ. 1), which only highlights how a model will perform on the data we have already collected, rather than providing guarantees about future performance for the infinite and continuous space that is the real-world. For the latter, we measured the worst-case failure rate for a variety of different model defenses (see Sect. 4), using several attacks that define the 'worst-case' according to different contexts (see Sect. 3). We trained one model for each dataset using the ResNet (He et al. 2015) architecture provided by Madry et al. as part of the "MNIST"[1] and "CIFAR-10"[2] challenges for each defense across several different hyperparameter combinations.

To generate confidence intervals, we varied both defense and attack parameters in an iterative grid search. Results across all tests are reported in Figures 4 and 8. Initial model weights and model architecture were taken from the survey by Madry et al Madry and Makelov (2017). Like in that paper (and commonly throughout the literature), we used the

[1] https://github.com/MadryLab/mnist_challenge.

[2] https://github.com/MadryLab/cifar10_challenge.

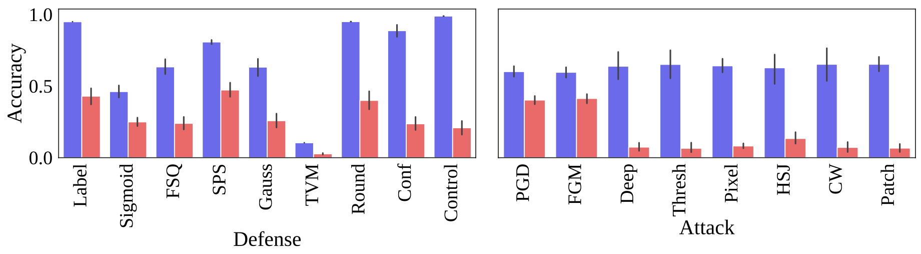

Fig. 12 The 95% confidence interval of adversarial (red) and benign (blue) accuracies for each defense (left) and each attack (right) for CIFAR-10. One trial was conducted for each hyper-parameter combination, and the confidence interval spans these trials

MNIST and CIFAR-10 datasets so that our survey can be directly compared to the wider literature. For each experiment, we trained the model for 20 epochs on the entire training set, as defined by the Tensorflow version of the datasets. Attacks were given a small computational budget of 10 iterations (100 samples, 10 iterations). For pre-processing defenses (see: Section), this included data transformations that were distinct from the original training process and for post-processing defenses, the model output varied (see: Section V-B2) relative to the survey by Madry et al. Madry and Makelov (2017). The MNIST dataset was classified using a simple toy model and the CIFAR-10 dataset was classified using a modified version of ResNet, both taken from the survey of Madry et al. Madry and Makelov (2017). Model prediction and attack times were measured using Python's 'process-time' due to the timing jitter associated with shared systems and operating system variability. The timing resolution was in milliseconds, far below the scale of training times and total attack times, making any noise negligible. After training, models were evaluated against the ten thousand unperturbed (benign) images available in the dataset according to both Equation 1 (accuracy) and Equation 2 (failure rate) in both the benign context. Each model was attacked using the same subset of 100 (disjoint) samples, randomly drawn from this set in an attempt to isolate model performance from coincidences associated with sampling. Model accuracy on this set of perturbed data is denoted as the 'adversarial accuracy' below, with 'benign accuracy' referring to the model accuracy on the unperturbed data. All experiments were run on an Intel Xeon 4210 with 32GB of memory and with an Nvidia V100 GPU in a shared-kernel environment. We evaluated hundreds of combinations of attacks and defenses as illustrated and enumerated in Fig. 11.

7 Results and discussions

7.1 Benign vs adversarial accuracy

Figure 12 (left subplot) depicts the 95% confidence region of adversarial and benign accuracy, computed using Eq. 1. The blue bars represent the accuracy on unperturbed (benign) data and the red bars represent the accuracy on perturbed (adversarial) data. From this rather large region (see Fig. 12), we see that both attack and defense hyper-parameter tuning have significant effects on the accuracy of a given method, since a small change in hyperparameters can drastically change the efficacy of a given attack or defense. Figure 11 shows the percent change in accuracy between the adversarial and benign circumstances for each defense and attack. We can plainly see that some techniques fool every model nearly every time while some attacks are weak and not likely to succeed under any conditions. Since no tested configuration reliably exceeded the *benign accuracy*, this metric seems to only indicate a lower bound of the generalization error while the *adversarial failure rate* estimates the upper bound (see Sect. 2.5). That is, we can confidently claim that our *true generalization error* (η) falls somewhere between 10^{-4} failures per second (roughly indicated by the 99.96 % test-set, benign accuracy for MNIST or 99.83% accuracy for CIFAR10) and the worst-case adversarial failure rate (roughly 10^{-1} for MNIST and 10^{2} for CIFAR-10), which obviously falls below the safety-critical standards (see Table 1) by huge margins, indicating that neither architecture is safety-critical (see: Sect. 5).

7.2 Defences

In Fig. 12 we see that, for every defense (left subplot), that adversarial accuracy is lower than benign accuracy, adding more empirical evidence the accuracy *vs.* robustness tradeoff discussed widely in the literature (e.g. there robustness and accuracy are at odds with each other). The tested defenses each attempt to strategically destroy, smooth, or average data in the original dataset or the output of the model, which results in a loss of precision that makes the benign accuracy worse than the Control (see: Fig. 12). Furthermore, when we examine the attacks in the right subplot of Fig. 12, we see that Deep, HSJ, CW, Pixel, Patch, and Thresh are all able to confuse the model more that half of the time. Furthermore, variations in the defense performance (right side of Figs. 20 and 14) raise questions about the ability of these architectures to generalize since things like bit-depth (FSQ), training-noise (Gausss-In), label-noise (Gauss-out), and image resolution (SPS) greatly vary the failure rate. In the real-world, effective resolution will change between individuals (e.g. a medical scan) or while moving (e.g. an autonomous vehicle). Even random noise drawn from approximately the same distribution as the training set (Gauss) increases the benign failure rate by an order of magnitude or two (compare the gap between *Gauss* and *Control* in both Figs. 14 and 20).

7.3 Attack

Attacks Fig. 12 (right subplot) depicts the same confidence region as above, but broken down by attack rather than defense. It is obvious that the Deep, HSJ, and Patch reduce the accuracy the most often. PGD, FGM, and CW are less effective, but still able to successfully perturb a significant portion of the samples. However, the Threshold and Pixel attacks are less consistent.Fig. 11 demonstrates how each attack fares against each defense by depicting the percent change in accuracy. The color gradient is centered at the benign failure rate, becoming a more intense red as the accuracy decreases, with a dark red indicating substantially worse performance than with the undefended model and unperturbed data. Since no column is blue in Fig. 11, no single defense is able to consistently subvert a generalized attacker, while modest gains against a particular attack are possible. Techniques like Adversarial Patch, DeepFool, and HopSkipJump consistently break models (see: Fig. 11). While some defenses do provide limited protection against more advanced techniques (indicated by the color blue in Fig. 11), their performance on unperturbed data tends to be reduced relative to the control (see: Fig. 12, left subplot) substantial empirical evidence that the normally discussed accuracy, from Eq. 1, is consistently more optimistic than what the adversarial analysis implies.

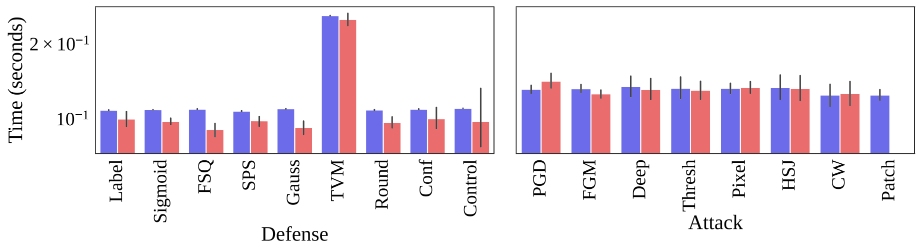

Fig. 13 The 95% confidence interval of prediction times (blue) and attack times (red) for CIFAR-10, broken down by defense (left) and attack (right). One trial is depicted for each hyper-parameter combination. Accuracy on the benign (unperturbed) dataset is depicted in blue and accuracy on the adversarial (perturbed) dataset is depicted in red

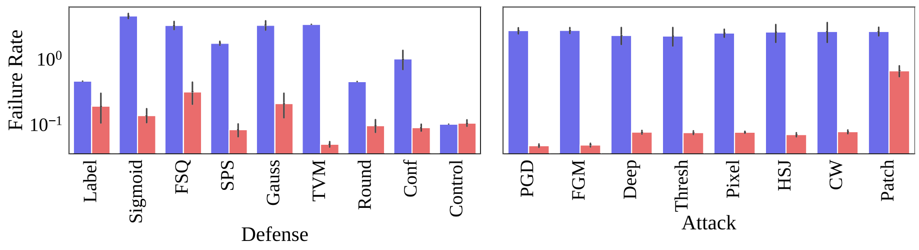

Fig. 14 The 95% confidence interval of failure rates both benign (blue) and adversarial (red) for CIFAR-10, broken down by defense (left) and attack (right). This was computed using Eq. 2. The failure rate of the benign (unperturbed) dataset is depicted in blue and the failure rate of the adversarial (perturbed) dataset is depicted in red

7.4 Computational cost

In order to estimate computational cost, we measured each time as a process time, reducing the jitter due to operating system operations and shared kernel constraints. We see that defenses require between 1 and 100 s of training per success, broken down by defense in the left subplot of Fig. 13. However, attacks (see right subplot of Fig. 13) require as few as several milliseconds per sample in the worst case and 10 seconds in the best case, with the average attack time falling after around a half second per sample. Furthermore, we see that the Fig. 14, we see how various defenses manage the trade-off between computational complexity and efficacy by measuring the failure rate as in Eq. 2. The Confident model had the best accuracy (see: Fig. 12) since it merely ignores queries below a certain confidence threshold, but even if we treat that as a null event (instead of a false classification), this increases the failure rate (see: Fig. 14) relative to the control since objects were not detected.

7.4.1 Failure rate

When we combine the information from the accuracy and time graphs in Eq. 2, we obtain Fig. 15, displaying the failure rate across attacks and defenses as well as their individualized performance.

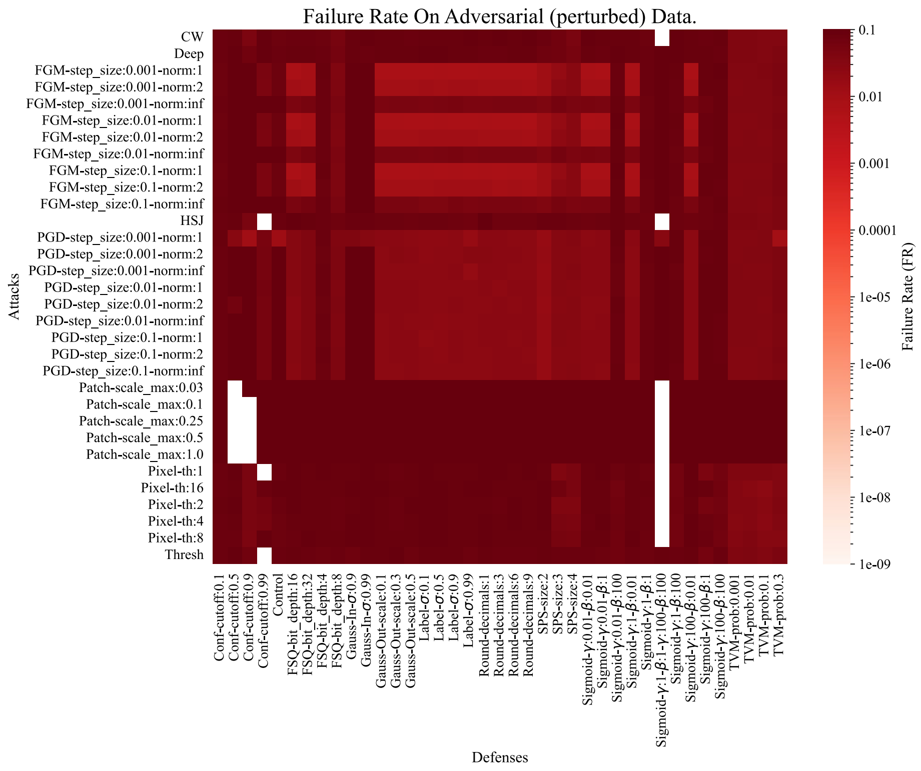

Fig. 15 The adversarial failure rate for all attacks and defenses for CIFAR-10. Darker red means worse performance and white indicates that no adversarial samples were found. This result is rather pessimistic, suggesting that all configurations fail to meet industrial standards

When we examine the average performance of a given defense (see: Fig. 15), we see that most defenses fail to meet safety critical standards (see: Table 1), even if the surveyed defenses tend to improve adversarial accuracy (see: Fig. 16). Total Variance Minimization produced particularly inaccurate models which led to particularly inaccurate attacks. While some defenses do provide relief against some attacks, that behavior is inconsistent across different attacks, particularly in the case of the Patch and Pixel attacks which seem to be universally strong, even with mild perturbation constraints. Furthermore, those gains are marginal compared to the efficacy of the average attack (see: Fig. 16) and the order of magnitude required by regulations (see: Fig. 1).

7.5 Attack budget and attacker knowledge

When we examine the general performance characteristics of attacks, we see that the average attack takes a few seconds to induce a failure (see Figs. 20 and 14). This appears to be consistent across attacks and is remarkably effective with only a small computational budget of 100 iterations and a query budget of 1000 with a perturbation distance no greater than 1 byte. This is true for both whitebox attacks (everything but HSJ) and blackbox (HSJ) attacks. Most attacks take a few seconds per sample; however, the one

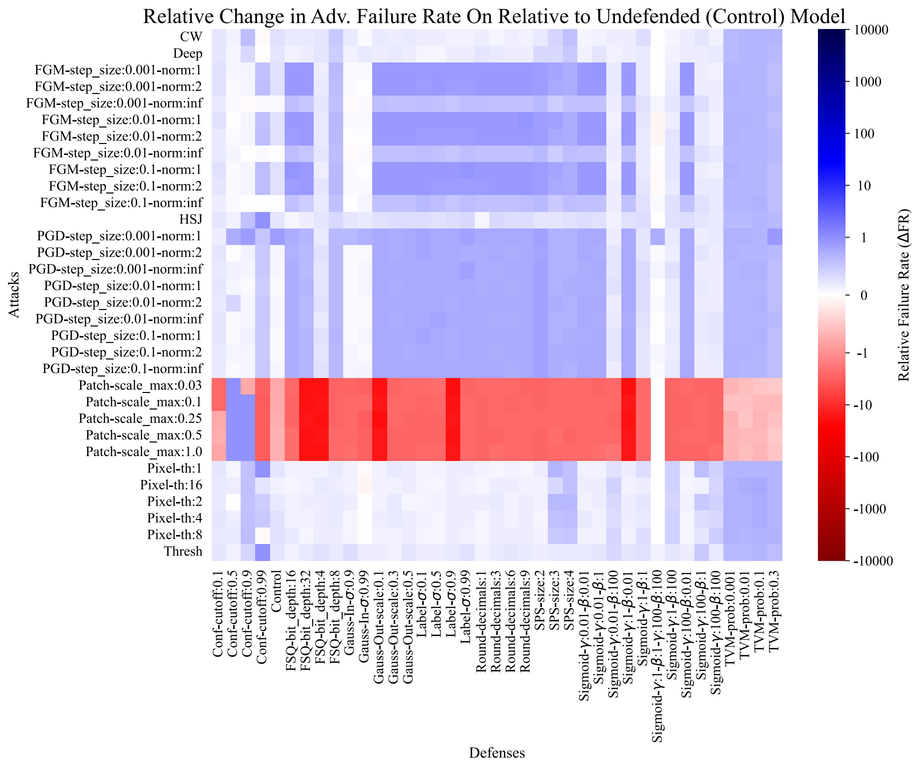

Fig. 16 The relative change in failure rate for the adversarial case, centered at the benign failure rate for CIFAR-10. Blue indicates an improvement relative to the benign (unperturbed) case. Red indicates a worsening of performance in the adversarial case compared to the benign case. White indicates no change. Note that the scale is logarithmic so that marginal gains and substantial losses can be seen clearly

exception to this is the Adversarial Patch attack which takes only a few milliseconds to induce a failure (on average). Due to the universality of this attack against an entire dataset (see Sect. 3) this failure rate would go to infinity as the sample size goes to infinity. In practice, a quadratic approximation of the boundary (HSJ) is roughly as effective as whitebox models, especially as the number of features scale (compare Fig. 11 and 18). However, there is a clear advantage when the attacker optimizes across a large sample of the test set (see *Patch* in Sect. 3), with the failure rate tending towards 0 as this sample size increases. Furthermore, by simulating a black-box attack (HSJ attack), regularizing generated examples for false confidence (CW attack), finding minimal separating planes (Deep attack), and using advanced optimization techniques to generate highly confident false examples (Thresh attack), we were able to consistently fool the models in mere milliseconds per sample. The simplest attacks, which Madry et al. proposed in 2017, remain effective (PGD, FGM). Furthermore, proper step-size tuning seems to compensate for this simplicity. This is evidenced by right side of Figs. 14 and 20, suggesting that the gradient ascent attacks might be a 'good enough' estimator of the generalized failure rate, especially given the astronomical gap between current test-set accuracy and regulatory requirements.

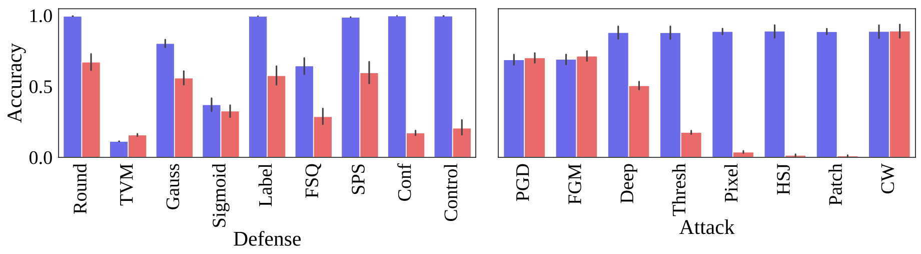

Fig. 17 The 95% confidence interval of adversarial and benign accuracies for each defense (left) and each attack (right) on the MNIST dataset, broken down by defense (left) and attack (right). One trial was conducted for each hyper-parameter combination

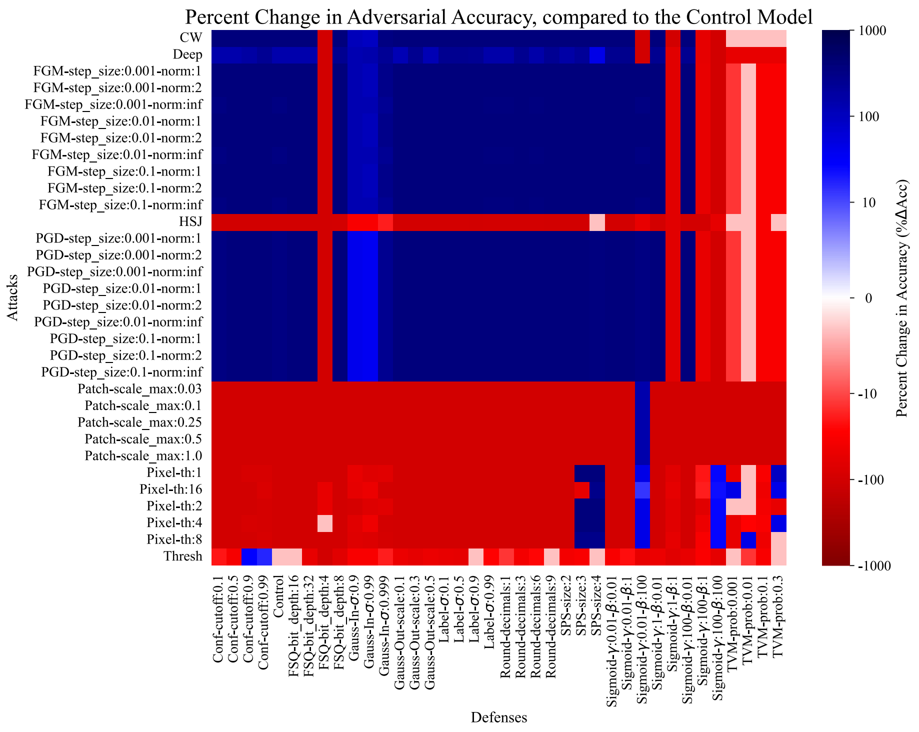

Fig. 18 The adversarial accuracy of each attack against each defense on the MNIST dataset. Blue would indicate an improvement relative to the undefended model. Red indicates that defense made a model worse. White indicates no change. This was computed using Eq. 3

7.6 Percent change in accuracy

Figure 11 demonstrates how each attack fares against each defense. The color gradient starts at white, indicating the undefended model's performance against unperturbed data, becoming a more intense red as the accuracy decreases, with blue indicating an increase in accuracy compared to the control model. As we can plainly see, no single defense is able to consistently subvert an attacker. While some defenses do provide limited protection against

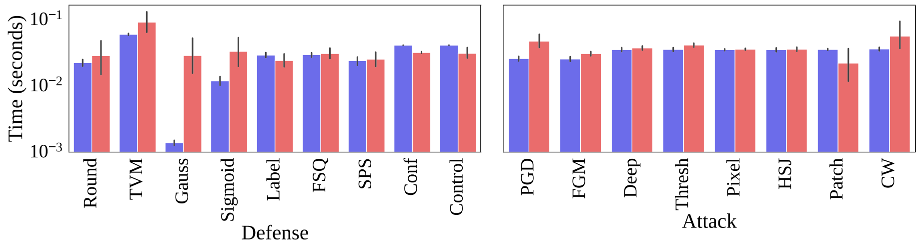

Fig. 19 The 95% confidence interval of prediction times (blue) and attack times (red) on the MNIST dataset, broken down by defense (left) and attack (right). Note that adversarial times are identical in each image since fitting and predicting are the same step in this case. One trial is depicted for each hyper-parameter combination

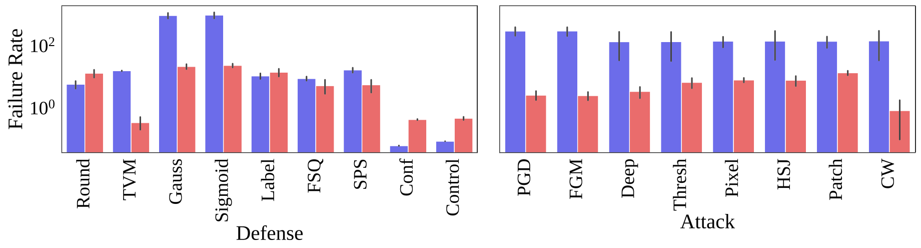

Fig. 20 The 95% confidence interval of failure rates, both benign (blue) and adversarial (red) on the MNIST dataset, broken down by defense (left) and attack (right). Note that no defense was able to improve upon the benign failure rate. This was computed using Eq. 2

more advanced techniques, their performance against gradient descent techniques is inconsistent at best. Furthermore, for DeepFool and HopSkipJump, we know that they could lead to even worse results for the defender, given a larger computational budget. While there is limited efficacy against gradient-based attacks for some defenses, advanced techniques like Adversarial Patch, DeepFool, and HopSkipJump consistently break models. That is, we provide substantial empirical evidence that the normally discussed accuracy from Eq. 1 is, at best, an optimistic estimate of the real-world failure rate.

7.7 Percent change in failure rate

Figure 16 depicts how each defense fares against each attack by comparing the change in failure rate when compared to the adversarial case on the undefended model. When we compare this plot to the one in Fig. 11, we see that, in many cases, the defenses were able to decrease the adversarial failure rate when compared to the undefended model, which is what these defenses intended to do. However, increased safety (depicted as blue in Fig. 16) is marginal at best—on the order of a few percent (see: Fig. 4) when measured rates are many orders of magnitude from regulatory standards (see: Table 1). Additionally, the potential downsides of a given defense are much larger (see: Fig. 11). This marginal improvement in the failure rate is driven largely by the marginal time cost (see: Fig. 13) rather than the accuracy (see: Fig. 12). That is, the defenses increase the adversarial accuracy, but also take significantly longer (see: Fig. 13).

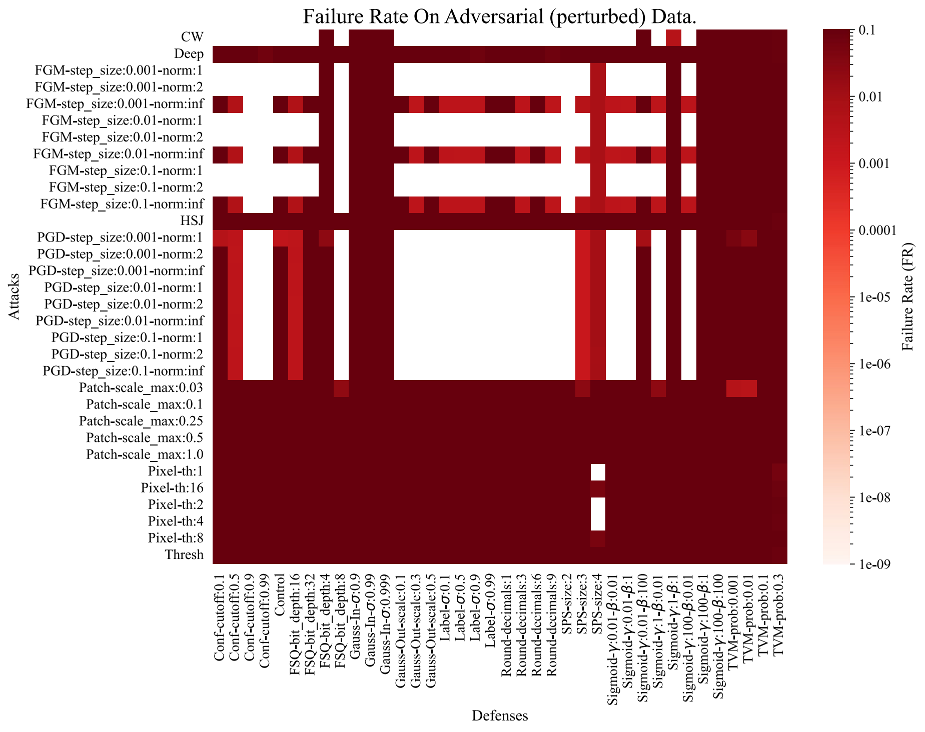

Fig. 21 The failure rate for the adversarial case on the MNIST dataset. A darker red indicates a worse performance and white indicates 0 induced failures. Note that the scale is logarithmic so that marginal gains and substantial losses can be seen clearly. White indicates cases where the attack was never successful

7.8 MNIST vs. CIFAR-10

In addition to running these experiments on the CIFAR-10 dataset, we also ran them on the MNIST dataset using a simpler model provided by Madry et al. Madry and Makelov (2017). Figure 17 broadly verifies the behavior observed above on the CIFAR-10, wherein defenses tend to reduce model accuracy on the benign set, gradient-based attack efficacy is largely determined by hyperparameter tuning, and non-gradient-based attacks are still very effective. As with CIFAR-10, no defense was able to improve the benign (unperturbed) accuracy of the undefended model for every attack (see: Fig. 18). Despite a smaller model and lower run-time requirements (compare Fig. 19 to Fig. 13), we see that training time is more-or-less the same. When we compare Fig. 15 with Fig. 21, we see that some of the defenses were much more effective at preventing gradient-descent attacks. However, they failed against the more computational expensive techniques of Patch, HSJ, and Deep. Then, when we examine failure rate (Fig. 20), we see that despite being a different dataset and model architecture, we are still able to induce failures in only a few seconds, as with CIFAR-10. However, when we break this rate down into each attack and defense combination and measure a change in the failure rate (see: Fig. 22), there seems to be a fairly consistent and marginal improvement driven by time (see: Fig. 19) rather than accuracy (see: Fig. 17), as with CIFAR-10. When we applied defenses to the simpler MNIST model and dataset (when compared to CIFAR-10), we found largely consistent results. However, we

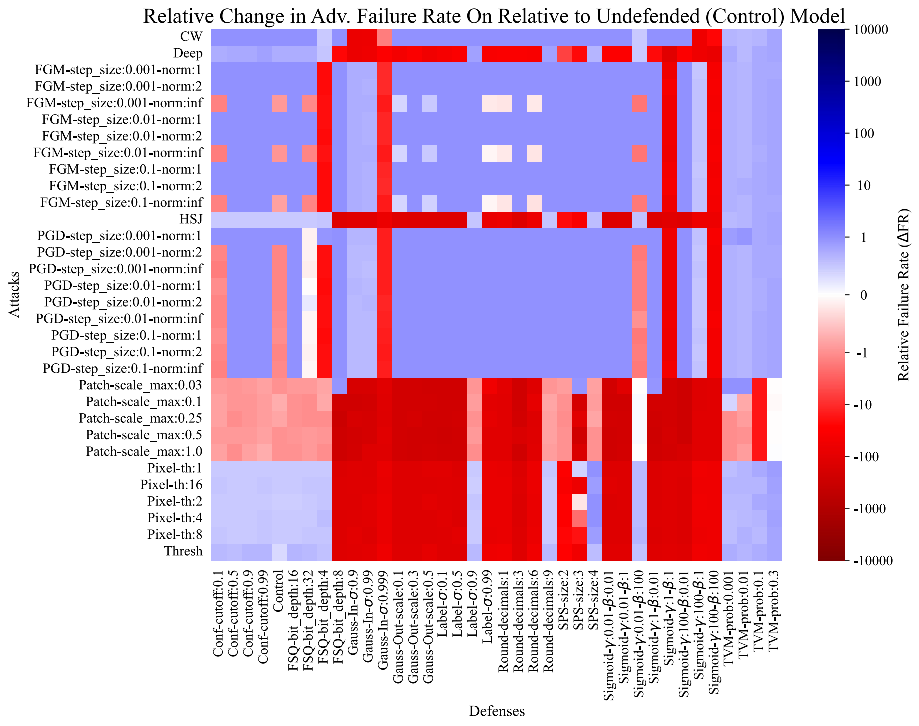

Fig. 22 The change in failure rate between the benign and adversarial cases on the MNIST dataset. Blue indicates an improvement relative to the undefended model. Red indicates that defense made a model worse. White indicates no change. Note that the scale is logarithmic so that marginal gains and substantial losses can be seen clearly. This was computed using Eq. 4

did find that there were more cases of failed adversarial attacks against MNIST, likely due to the significantly smaller dimensionality of both the dataset and model (Dohmatob 2019). However, we would expect real-world data to be significantly higher resolution and full-color, unlike the black and white low-resolution images typical of MNIST, so those results probably underestimate the severity of the problem in real-world systems that use multiple multi-pixel RGB cameras to classify objects in real-time.

8 Limitations

8.1 True failure rate estimation

While it is true that real-world noise can inadvertently become adversarial, it is obvious that not every possible noise vector will increase the loss for a given sample. We can, however, confidently say that the *true generalization error* lies between the test-set accuracy (Eq. 1) and the adversarial failure rate. Further work remains regarding the gap between these two estimates, but falls outside the scope of this paper. However, as we show in the results (Sect. 7) this fact hardly matters in the context of modern computer vision models, because **both of these measures** fail to meet safety-critical standards (see Table 1) by many orders of magnitude (see Fig. 20 and Fig. 14.)

8.2 On optimal attacks

Additionally, while these attacks are quite efficient, none are provably the fastest possible attack. So, *at best*, they underestimate the failure rate. In the case of our safety-critical analysis, this amplifies and does not diminish our claims, raising serious concerns about using in-distribution test data as an indicator of real-world performance. Therefore, because these attacks are not provably optimal, this failure rate should not be taken as an absolute measure of the true failure rate. However, it is still a reliable metric for comparing the efficacy of two models as evidenced by the relatively consistent failure rates across various defenses and hyperparameter configurations for a given attack (right side of Figs. 20 and 14).

8.3 Model selection

In general, we did not choose the model architecture, but relied on two reference models provided by Madry et al. They have been tested and cited numerous times (Madry and Makelov 2017). The point of this work is not to chase state-of-the-art results, but to evaluate robustness-maximizing techniques in a controlled manner while highlighting useful metrics and techniques for doing so. At the time of publishing, the authors are not aware of *any* architecture that meets safety critical standards when measured test-set accuracy sense (see Eq. 1) and there's no demonstrated technique (to the knowledge of the authors) to reduce this in the adversarial sense without sacrificing accuracy or computational time in the benign case (see Equ. 2, Figs. 14 and 20). So the general conclusion about the real-time safety-critical nature of modern neural networks would remain the same for any other architecture known to the authors.

8.4 On attacker's knowledge

In this paper, we have tested white-box attacks (FGM, PGD, CW), attacks that need access to the model probability outputs (Deep, Pixel, Thresh, Patch), and a single black-box method (HSJ), outlined in Sect. 3. On one hand, it would seemingly be unfair to compare these methods under the same constraints. However, as we show in the results (Sect. 7), the apparent advantage of attackers with more knowledge of the model tends to disappear under the added computational burden. That is, the HSJ attack (Sect. 3 and Fig. 10), which only relies on hard class-labels and an offline model approximation, is consistently effective at fooling models while also outperforming attacks that have access to more information (see: Figs 3, 4, 5, 6, 7, 8, 9). The interesting question, then, isn't necessarily how good of an adversarial sample one can generate, but *the rate* at which *any* misclassification can be induced. This *worst-case failure rate* will define the feasible upper bound on a target hardware architecture. Since these attacks vary substantially in run-time and information requirements, current methods relying on accuracy measures do not distill the efficacy of a given attack or defence from the perspective of any model-builder or attacker with a fixed computational budget. However, by controlling for the number of queries and normalizing by CPU-time (see: Sect. 6), we are able to isolate the effect of defence techniques against a wide-variety of idealized attackers that optimize for different distance metrics and are subject to very different constraints. By no means do we want to minimize the offline possibilities (see: Sect. 3.3) of attacks like "Adversarial Patch" and "Hop Skip Jump". Instead, we seek to highlight the computational triviality of these attacks and raise sincere questions about the safety of these models in general.

9 Conclusions

Neural networks are being deployed in a wide variety of industrial applications with real-world safety considerations, that despite high accuracy scores, fail against a wide variety of attacks that overwhelmingly require fewer computational resources than building the original model. While 'weak' attacks are fast, they require hyper-parameter tuning and retrospective evaluations that make them less effective, but nonetheless cheap-enough to execute, requiring only a few seconds of CPU time. The efficacy of even single-byte or single-pixel attacks against otherwise very accurate models raises questions not only about the intentional adversary, but also how a system will handle real-world, 'legitimate' anomalies like dust, lens aberration, and sensor failure.

When we consider the attack that finds the minimal class-separating distance, DeepFool, we see that 80% of all samples are corruptible under our meager one-byte distance constraint. When we remove this distance constraint, we find that nearly every sample can be fooled with the addition of an adversarial 'patch' on the original image (Fig. 11) even when we constrain an attack to a small number of iterations (≤ 10). That is, if we assume that our attacker has an unlimited budget, we cannot even hope to defend against these attacks. However, the relative ease of finding such adversarial examples suggests that these attacks provide an empirical estimate of a 'worst case' failure rate (see Eq. 2) in their respective contexts. Therefore, adversarial model analysis provides a computationally cheap way to analyze the worst case failure rate of a system without having to collect, label, perturb, and predict many thousands of test images.

Furthermore, since this failure rate is based on process time, it is obvious that more powerful hardware, without underlying changes to the model architecture, would result in a larger failure rate since we merely fail on more samples in the same amount of time. That is to say, these problems are inherent to the model and not something that we can solve with more processor cycles. Furthermore, even when we obscure everything from the user except model output, we can consistently break models in as few as ten queries to a hard class-label API (e.g., by using the HopSkipJump attack). We found that adding Gaussian noise to the model outputs or training images, reducing the bit-depth of the numerical calculations to match the precision of the image, and setting a confidence threshold were marginally effective defenses when compared to the control model. However, none of these defenses reach the regulatory standards required for safety-critical systems—in fact, they fail by several orders of magnitude. Furthermore, any gains seen in the failure rate (Fig. 16) are inconsistent while universally reducing the accuracy on the original (unperturbed/benign) data (Fig. 11). Improvements due to the defenses are rare and marginal, and they consistently make benign performance worse (see Figs. 11 and 18). We find this to be true across both datasets and model architectures. Since we know that model accuracy scales with $\mathcal{O}(1/\sqrt{n})$ (Vapnik et al. 1994) and attacks seem to run in constant time (see Figs. 13 and 19) for a fixed computational and query budgets of a few seconds and 1000 queries respectively.

Finally, due to the rate at which we can generate feasible misclassifications, we must question the real-world efficacy of any system that relies on these models to make predictions that meet the legal standards that define 'safe'. Despite this pessimism, adversarial model analysis proves to be a computationally efficient way to analyze and compare the out-of-distribution robustness of model architectures without the need to generate massive test sets from real world data.

Acknowledgements Financial support has been provided in part by the Knut and Alice Wallenberg Foundation grant no. 2019.0352 and by the eSSENCE Programme under the Swedish Government's Strategic Research Initiative.

Funding Open access funding provided by Umea University.

References

Al-Qizwini M, Barjasteh I, Al-Qassab H, Radha H (2017) Deep learning algorithm for autonomous driving using GoogLeNet. In: 2017 IEEE Intelligent Vehicles Symposium (IV), 89–96. IEEE

Aljuhani A (2021) Machine learning approaches for combating distributed denial of service attacks in modern networking environments. IEEE Access 9:42236–42264

Anselm B, Andrzej E, David H, Warmuth Manfred K (1989) Learnability and the vapnik-chervonenkis dimension. J ACM 36(4):929–965

Athalye A, Carlini N, Wagner D (2018) Obfuscated gradients give a false sense of security: circumventing defenses to adversarial examples. arXiv:1802.00420 [cs], July

Banks VA, Plant KL, Stanton NA (2018) Driver error or designer error: using the perceptual cycle model to explore the circumstances surrounding the fatal tesla crash on 7 May 2016. Safety Sci 108:278–285

Battista B, Giorgio F, Fabio R (2009) Multiple classifier systems for adversarial classification tasks. In: Benediktsson JA, Kittler J, Roli F (eds) Multiple classifier systems, lecture notes in computer science. Springer, Berlin, pp 132–141

Bect J, Li L, Vazquez E (2017) Bayesian subset simulation. SIAM/ASA J Uncertain Quantif 5(1):762–786

Bernal G, Colombo S, Al Ai Baky M, Casalegno F 2017 Safety++ designing IoT and wearable systems for industrial safety through a user centered design approach. In: Proceedings of the 10th international conference on pervasive technologies related to assistive environments, pp 163–170

Biggio B, Corona I, Maiorca D, Nelson B, Šrndić N, Laskov P, Giacinto G, Roli F, (2013) Evasion Attacks against machine learning at test time. arXiv:1708.06131 [cs], 7908: 387–402

Biggio B, Corona I, Maiorca D, Nelson B, Srndic N, Laskov P, Giacinto G, Roli F (2013) Evasion attacks against machine learning at test time. arXiv:1708.06131 [cs], 7908: 387–402

Bloom C, Tan J, Ramjohn J, Bauer L (2017) Self-driving cars and data collection: privacy perceptions of networked autonomous vehicles. In: Symposium on usable privacy and security (SOUPS)

Brown TB, Mané D, Roy A, Abadi M, Gilmer (2017) J adversarial patch. arXiv:1712.09665

Buolamwini J, Gebru T (2018) Gender shades: intersectional accuracy disparities in commercial gender classification. In: Conference on fairness, accountability and transparency, pp 77–91, PMLR

Carlini N, Wagner D (2017) Towards evaluating the robustness of neural networks. arXiv:1608.04644 [cs], March

Chae H, Kang CM, Kim BD, Kim J, Chung CC, Choi JW (2017) Autonomous braking system via deep reinforcement learning. In: IEEE 20th international conference on intelligent transportation systems (ITSC)

Chakraborty A, Alam M, Dey V, Chattopadhyay A, Mukhopadhyay D (2018) Adversarial attacks and defences: a survey. arXiv:1810.00069 [cs, stat]

Chambolle A (2004) An algorithm for total variation minimization and applications. J Math Imag Vision 20(1):89–97

Chen J, Jordan MI, Wainwright MJ (2020) HopSkipJumpAttack: a query-efficient decision-based attack. In IEEE symposium on security and privacy (sp), IEEE, pp 1277–1294

Ching T, Himmelstein Daniel S, Beaulieu-Jones Brett K, Kalinin Alexandr A, Do Brian T, Way Gregory P, Ferrero E, Agapow PM, Zietz M, Hoffman Michael M, Xie W, Rosen Gail L, Lengerich Benjamin J, Israeli J, Lanchantin J, Woloszynek S, Carpenter Anne E, Shrikumar Avanti X, Evan JC, Lavender Christopher A, Turaga Srinivas C, Alexandari Amr M, Lu Laura K, Segler Marwin HSB, Swamidass SJ, Huang A, Anthony G, Casey SG (2017) Opportunities and obstacles for deep learning in biology and medicine. J R Soc Interface 15(141):20170387

Cintas C, Speakman S, Akinwande V, Ogallo W, Weldemariam K, Sridharan S, McFowland E (2020) Detecting adversarial attacks via subset scanning of autoencoder activations and reconstruction error. In: Proceedings of the twenty-ninth international joint conference on artificial intelligence, Yokohama, pp 876–882

Colbrook MJ, Antun V , Hansen AC 2021 Can stable and accurate neural networks be computed. On the barriers of deep learning and Smale's 18th problem. arXiv, 2101

Corsaro William A (1982) Something old and something new: the importance of prior ethnography in the collection and analysis of audiovisual data. Sociol Methods Res 11(2):145–166

Cosentino J, Zaiter F, Pei D, Zhu J (2019) The search for sparse, robust neural networks. arXiv:1912.02386

Croce F, Hein M (2020) Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. arXiv:2003.01690[cs, stat], August

Daya AA, Salahuddin MA, Limam N, Boutaba R (2019) A graph-based machine learning approach for bot detection. arXiv:1902.08538 [cs], February

Deborah R, Jack M, Farrell A (2000) A resource guide on racial profiling data collection system: promising practices and lessons learned. US Department of Justice, Washington D C

Desislavov R, Martínez-Plumed F, Hernández-Orallo J (2021) Compute and energy consumption trends in deep learning inference. arXiv:2109.05472

Dohmatob E (2019) Generalized no free lunch theorem for adversarial robustness. In: Proceedings of the 36th international conference on machine learning, 97 of PMLR

Emre P (2019) Artificial intelligence in radiology: friend or foe? where are we now and where are we heading? Acta Radiol Open 8(2):2058460119830222

Finlayson SG, Chung HW, Kohane IS, Beam AL (2018) Adversarial attacks against medical deep learning systems. arXiv:1804.05296

Fredrikson M, Jha S, Ristenpart T (2015) Model Inversion attacks that exploit confidence information and basic countermeasures. In: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security - CCS '15, ACM Press, Colorado, pp 1322–1333

Fukuda T, Shibata T (1992) Theory and applications of neural networks for industrial control systems. IEEE Trans Ind Electron 39(6):472–489

Gichoya JW, Banerjee I, Bhimireddy AR, Burns J, Celi LA, Chen L-C, Correa Ramon, Dullerud N, Ghassemi M, Huang S-C et al (2022) Ai recognition of patient race in medical imaging: a modelling study. The Lancet Digital Health 4(6):e406–e414

Goodfellow IJ, Shlens J, Szegedy C (2014) Explaining and harnessing adversarial examples. arXiv:1412.6572

Grigorescu S, Trasnea B, Cocias T, Macesanu G (2020) A survey of deep learning techniques for autonomous driving. J Field Robot 37(3):362–386

Hadj-Selem F, Löfstedt T, Dohmatob E, Frouin V, Dubois M, Guillemot V, Duchesnay E (2018) Continuation of nesterov's smoothing for regression with structured sparsity in high-dimensional neuroimaging. IEEE Trans Med Imag 37(11):2403–2413

He K, Zhang X, Ren S, Sun J (2015) Deep residual learning for image recognition. CoRR, arXiv:abs/1512.03385

ICOH. Global estimates of occupational accidents and work-related illnesses 2017. International commission on occupational health (2017)

International Electrotechnical Commission. IEC 62304 medical device software–software life cycle processes. International electrotechnical commission, 2nd edition, 2006

International Electrotechnical Commission. IEC 61508 safety and functional safety. International electrotechnical commission, 2nd edition, 2010

International Standards Organization. (2018) ISO 26262-1:2011, road vehicles—functional safety. https://www.iso.org/standard/43464.html (visited 2022-04-20)

Jakubovitz D, Giryes R (2018) Improving dnn robustness to adversarial attacks using jacobian regularization. In: Proceedings of the European conference on computer vision (ECCV), 514–529

Jeffrey H, Jill P, Silverman Joseph H (2010) Complexity theory and P vs NP. Springer, Berlin, pp 258–262

Jian T, Wang Z, Wang Y, Dy J, Ioannidis S (2022) Pruning adversarially robust neural networks without adversarial examples. arXiv:2210.04311

Koch B, Denton E, Hanna A, Foster JG (2021) Reduced, reused and recycled: the life of a dataset in machine learning research. arXiv preprint arXiv:2112.01716

Kotyan S, Vargas DV (2019) Adversarial robustness assessment: why both l_0 and l_∞ attacks are necessary. arXiv e-prints, pages arXiv–1906,

Lam H (2004) New design-to-test software strategies accelerate time-to-market. In IEEE/CPMT/SEMI 29th international electronics manufacturing technology symposium (IEEE Cat. No. 04CH37585), IEEE, 140–143

Leanna R (2017) Developing safety-critical software: a practical guide for aviation software and DO-178C compliance. CRC Press, Boca Raton

Lecuyer M, Atlidakis V, Geambasu R, Hsu D, Jana S (2019) Certified robustness to adversarial examples with differential privacy. In: 2019 IEEE symposium on security and privacy (SP), 656–672

Lee T, Edwards B, Molloy IM, Su D (2018) Defending against model stealing attacks using deceptive perturbations. CoRR, arXiv: abs/1806.00054

Leonard E, Gerrish Peter H (2001) Gender and age influence on fatality risk from the same physical impact determined using two-car crashes. SAE Trans 110:1336–1341

Li Chen, Jun Xiao, Zou Pu, Haifeng Li (2021) Lie to me: a soft threshold defense method for adversarial examples of remote sensing images. IEEE Geosci Remote Sens Lett 19:1–5

Li B, Vorobeychik Y, Chen X (2016) A general retraining framework for scalable adversarial classification. arXiv:1604.02606[cs, stat], November

Lu K, Mardziel P, Wu F, Amancharla P, Datta A (2020) Gender bias in neural natural language processing. logic, language, and security: essays dedicated to andre scedrov on the occasion of his 65th birthday, pp 189–202

Madry A, Makelov A, Ludwig S, Dimitris T, Adrian V (2017) Towards deep learning models resistant to adversarial attacks. arXiv:1706.06083

Makary Martin A, Michael D (2016) Medical error-the third leading cause of death in the US. BMJ. https://doi.org/10.1136/bmj.i2139

Miller DJ, Xiang Z, Kesidis G (2020) Adversarial learning targeting deep neural network classification: a comprehensive review of defenses against attacks. Proceed IEEE 108(3):402–433

Monmasson E, Idkhajine L, Cirstea MN, Bahri I, Tisan Alin, Naouar MohamedWissem (2011) Fpgas in industrial control applications. IEEE Trans Ind Inform 7(2):224–243

Moosavi-Dezfooli S-M, Fawzi A, Frossard P (2016) Deepfool: a simple and accurate method to fool deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2574–2582

National Highway Transportation Safety Administration's (NHTSA) National center for statistics and analysis. Critical reasons for crashes investigated in the national motor vehicle crash causation survey. https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812115 (visited 2022-04-20), 2015

Nelson BA (2010) Behavior of machine learning algorithms in adversarial environments. University of California, Berkeley

Park J, Nyang D, MA (2018) Timing is almost everything: realistic evaluation of the very short intermittent ddos attacks. In: 2018 16th annual conference on privacy, security and trust (PST), pp 1–10

Paudice A, Muñoz-González L, Gyorgy A, Lupu EC (2018) Detection of adversarial training examples in poisoning attacks through anomaly detection. arXiv:1802.03041[cs, stat], February

Pearson RK (2005) Mining imperfect data: dealing with contamination and incomplete records. SIAM

Roh Y, Heo G, Whang SE (2019) A survey on data collection for machine learning: a big data-ai integration perspective. IEEE Trans Knowl Data Eng 33(4):1328–1347

Ross A, Doshi-Velez F (2018) Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In: Proceedings of the AAAI conference on artificial intelligence, p 32

Rudin L, Osher S, Fatemi E (1992) Nonlinear total variation based noise removal algorithms. Physica D: Nonlinear Phenomena 60(1–4):259–268

Sahiner B, Pezeshk A, Hadjiiski LM, Wang X, Drukker K, Cha KH, Summers RM, Giger M (2019) Deep learning in medical imaging and radiation therapy. Med Phys 46(1):e1–e36

Sehwag V, Wang S, Mittal P, Jana S. (2019) Towards compact and robust deep neural networks. arXiv:1906.06110

Shokri R, Stronati M, Song C, Shmatikov V (2017) Membership inference attacks against machine learning models. In: 2017 IEEE symposium on security and privacy (SP), IEEE, 3–18

Sinn M, M W, B B, MI N, M T (2019). Evolutionary search for adversarially robust neural networks, In safe machine learning workshop at ICLR

SrinivasAcharyulu PV, Seetharamaiah P (2015) A framework for safety automation of safety-critical systems operations. Saf Sci 77:133–142

The Organisation for Economic Co-operation and Development. OECD statistics. https://stats.oecd.org/ (visited 2022-04-20), (2020)

Tramèr F, Papernot N, Goodfellow I, Boneh D, McDaniel P (2017). The space of transferable adversarial examples. arXiv:1704.03453

Tramèr F, Zhang F, Juels A, Reiter MK, Ristenpart T (2016) Stealing machine learning models via prediction APIs. In: 25th USENIX security symposium (USENIX Security 16), 601–618

Tsipras D, Santurkar , Engstrom L, Turner A, Madry A (2019). Robustness may be at odds with accuracy. arXiv:1805.12152[cs, stat], September

Tuan LA, Zheng MC, Tho QT (2010) Modeling and verification of safety critical systems: a case study on pacemaker. In: 2010 fourth international conference on secure software integration and reliability improvement, IEEE, pp 23–32

Vapnik V, Levin E, Le Cun Y (1994) Measuring the vc-dimension of a learning machine. Neural Comput 6(5):851–876

von Ahn L, Blum M, Hopper NJ, Langford J (2003) Captcha: using hard ai problems for security. In: International conference on the theory and applications of cryptographic techniques, Springer, pp 294–311

Wang X, Li J, Kuang X, Tan Y, Li Jin (2019) The security of machine learning in an adversarial setting: a survey. J Parallel Distrib Comput 130:12–23

Warde-Farley D, Goodfellow I (2017) Adversarial perturbations of deep neural networks. In: Tarlow D, Hazan T, Papandreou G (eds) Perturbations, Optimization, and Statistics. The MIT Press, Cambridge

Xiao Z, Gao X, Fu C, Dong Y, Gao W, Zhang X, Zhou J, Zhu J (2021) Improving transferability of adversarial patches on face recognition with generative models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 11845–11854

Xu W, Evans D, Qi Y (2017) Feature squeezing: detecting adversarial examples in deep neural networks. arXiv:1704.01155

Zantedeschi V, Nicolae M-I, Rawat A (2017) Efficient defenses against adversarial attacks. In: Proceedings of the 10th ACM workshop on artificial intelligence and security, AISec '17, association for computing machinery, New York, pp 39–49

Zhang Y, Liang P (2019) Defending against whitebox adversarial attacks via randomized discretization. In: The 22nd international conference on artificial intelligence and statistics, PMLR, 684–693

Zirger BJ, Hartley JL (1996) The effect of acceleration techniques on product development time. IEEE Trans Eng Manag 43(2):143–152