Characterizing Data Point Vulnerability via Average-Case Robustness

Tessa Han*[1] **Suraj Srinivas***[1] **Himabindu Lakkaraju**[1]

[1]Harvard University

Abstract

Studying the robustness of machine learning models is important to ensure consistent model behaviour across real-world settings. To this end, adversarial robustness is a standard framework, which views robustness of predictions through a binary lens: either a worst-case adversarial misclassification exists in the local region around an input, or it does not. However, this binary perspective does not account for the degrees of vulnerability, as data points with a larger number of misclassified examples in their neighborhoods are more vulnerable. In this work, we consider a complementary framework for robustness, called average-case robustness, which measures the fraction of points in a local region that provides consistent predictions. However, computing this quantity is hard, as standard Monte Carlo approaches are inefficient especially for high-dimensional inputs. In this work, we propose the first analytical estimators for average-case robustness for multi-class classifiers. We show empirically that our estimators are accurate and efficient for standard deep learning models and demonstrate their usefulness for identifying vulnerable data points, as well as quantifying robustness bias of models. Overall, our tools provide a complementary view to robustness, improving our ability to characterize model behaviour.

1 INTRODUCTION

A desirable attribute of machine learning models is robustness to perturbations of input data. A popular notion of robustness is adversarial robustness, the ability of a model to maintain its prediction when presented with adversarial perturbations, i.e., perturbations designed to cause the model to change its prediction. Although adversarial robustness identifies whether a misclassified example exists in a local region around an input, it fails to capture the degree of vulnerability of that example, indicated by the difficulty in finding an adversary. For example, if the model geometry is such that 99% of the local region around an example (say, point A) contains correctly classified examples, this makes it harder to find an adversarial example as compared to the case where only 1% of the local region (say, for point B) contains correctly classified examples, where even random perturbations may be misclassified. However, from the adversarial robustness perspective, the prediction at a point is declared either robust or not, and thus both points A and B are considered equally non-robust (see Figure 1 for an illustrative example). The ease of obtaining a misclassification, or *data point vulnerability*, is captured by another kind of robustness: *average-case robustness*, i.e., the fraction of points in a local region around an input for which the model provides consistent predictions. * If this fraction is less than one, then an adversarial perturbation exists. The smaller this fraction, the easier it is to find a misclassified example. While adversarial robustness is motivated by model security, average-case robustness is better suited for model and dataset understanding, and debugging.

Standard approaches to computing average-case robustness involve Monte-Carlo sampling, which is computationally inefficient especially for high-dimensional data. For example, Cohen et al. [2019] use $n = 100,000$ Monte Carlo samples per data point to compute this quantity. In this paper, we propose to compute average-case robustness via

*Equal contribution

*In addition to the size of the misclassified region, another factor that affects the ease of finding misclassified examples is the specific optimization method used. In this study, we aim to study model robustness in a manner agnostic to the specific optimization used, and thus, we only focus on the size of the misclassified region. We believe this study can form the basis for future studies looking into the properties (e.g., "ease of identifying misclassified examples") of specific optimization methods.

Proceedings of the 40th Conference on Uncertainty in Artificial Intelligence (UAI 2024), PMLR 244:1514–1541.

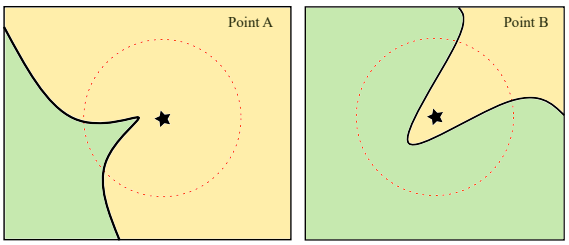


Figure 1: Consider a binary classifier (green vs. yellow) and points A (left) and B (right), both correctly classified to the yellow class. The dotted red circles represent ϵ-balls around the data points. Although adversarial robustness rightly considers the model non-robust at both points (due to the existence of adversarial examples within the ϵ-ball), it fails to discern that point B has a larger fraction of misclassified points in its neighborhood, making it more vulnerable than point A, an aspect exactly captured by average-case robustness.

analytical estimators, reducing the computational burden, while simultaneously providing insight into model decision boundaries. Our estimators are exact for linear models and well-approximated for non-linear models, especially those having a small local curvature [Moosavi-Dezfooli et al., 2019, Srinivas et al., 2022]. Overall, our work makes the following contributions:

1. We derive novel analytical estimators to efficiently compute the average-case robustness of multi-class classifiers. We also provide estimation error bounds for these estimators that characterizes approximation errors for non-linear models.
2. We empirically validate our analytical estimators on standard deep learning models and datasets, demonstrating that these estimators accurately and efficiently estimate average-case robustness.
3. We demonstrate the usefulness of our estimators in two case studies: identifying vulnerable samples in a dataset and measuring class-level robustness bias [Nanda et al., 2021], where we find that standard models exhibit significant robustness bias among classes.

To our knowledge, this work is the first to investigate analytical estimation of average-case robustness for the multi-class setting. In addition, the efficiency of these estimators makes the computation of average-case robustness practical, especially for large deep neural networks.

2 RELATED WORK

Adversarial robustness. Prior works have proposed methods to generate adversarial attacks [Carlini and Wagner, 2017, Goodfellow et al., 2015, Moosavi-Dezfooli et al., 2016], which find adversarial perturbations in a local region around a point. In contrast, this work investigates average-case robustness, which calculates the probability that a model's prediction remains consistent in a local region around a point. Prior works have also proposed methods to certify model robustness [Cohen et al., 2019, Carlini et al., 2022], guaranteeing the lack of adversarial examples for a given ϵ-ball under certain settings. Specifically, Cohen et al. [2019] propose randomized smoothing, which involves computing class-wise average-case robustness, which is taken as the output probabilities of the randomized smoothed model. However, they estimate these probabilities via Monte Carlo sampling with $n = 100,000$ samples, which is computationally expensive. Viewing average-case robustness from the lens of randomized smoothing, our estimators can also be seen as providing an analytical estimate of randomized smoothed models. However, in this work, we focus on their applications for model understanding and debugging as opposed to improving robustness.

Probabilistic robustness. Prior works have explored notions of probabilistic and average-case robustness. For instance, Fazlyab et al. [2019], Kumar et al. [2020], Mangal et al. [2019] focus on certifying robustness of real-valued outputs to input perturbations. In contrast, this work focuses on only those output changes that cause misclassification. Like our work, Franceschi et al. [2018] also considers misclassifications. However, Franceschi et al. [2018] aims to find the smallest neighborhood with no adversarial example, while we compute the probability of misclassification in a given neighborhood. Robey et al. [2022], Rice et al. [2021] also aim to compute average-case robustness. However, they do so by computing the average loss over the neighborhood, while we use the misclassification rate. Closest to our work is the work by Weng et al. [2019] which aims to certify the binary misclassification rate (with respect to a specific class to misclassify to) using lower and upper linear bounds. In contrast, our work estimates the multi-class misclassification rate, as opposed to bounding the quantity in a binary setting. A crucial contribution of our work is its applicability to multi-class classification and the emphasis on estimating, rather than bounding, robustness.

Robustness to distributional shifts. Prior works have explored the performance of models under various distributions shifts [Taori et al., 2020, Ovadia et al., 2019]. From the perspective of distribution shift, average-case robustness can be seen as a measure of model performance under Gaussian noise, a type of natural distribution shift. In addition, in contrast to works in distributional robustness which seek to build models that are robust to distributions shifts [Thulasidasan et al., 2021, Moayeri et al., 2022], this work focuses on measuring the vulnerability of existing models to Gaussian distribution shifts.

3 AVERAGE-CASE ROBUSTNESS ESTIMATION

In this section, we first describe the mathematical problem of average-case robustness estimation. Then, we present the naïve estimator based on Monte Carlo sampling and derive more efficient analytical estimators.

3.1 NOTATION AND PRELIMINARIES

Assume that we have a neural network $f : \mathbb{R}^d \to \mathbb{R}^C$ with C output classes and that the classifier predicts class $t \in [1, ..., C]$ for a given input $\mathbf{x} \in \mathbb{R}^d$, i.e., $t = \arg\max_{i=1}^{C} f_i(\mathbf{x})$, where f_i denotes the logits for the i^{th} class. Given this classifier, the average-case robustness estimation problem is to compute the probability of consistent classification (to class t) under noise perturbation of the input.

Definition 1. *We define the **average-case robustness** of a classifier f to noise $\mathcal{R}$ at a point $\mathbf{x}$ as*

$$p^{\text{robust}}(\mathbf{x}, t) = P_{\epsilon \sim R}\left[\arg\max_i f_i(x + \epsilon) = t\right]$$

The more robust the model is in the local neighborhood around $\mathbf{x}$, the larger the average-case robustness measure $p^{\text{robust}}(\mathbf{x}, t)$. In this paper, given that robustness is always measured with respect to the predicted class t at $\mathbf{x}$, we henceforth suppress the dependence on t in the notation. We also explicitly show the dependence of p^{robust} on the noise scale σ by denoting it as p^{robust}_σ.

In this work, we shall consider $\mathcal{R}$ as an isotropic Normal distribution, i.e., $\mathcal{R} = \mathcal{N}(0, \sigma^2)$. However, as we shall discuss in the next section, it is possible to accommodate both non-isotropic and non-Gaussian distributions in our method. Note that for high-dimensional data ($d \to \infty$), the isotropic Gaussian distribution converges to the uniform distribution on the surface of the sphere with radius $r = \sigma\sqrt{d}$ [†] due to the concentration of measure phenomenon [Vershynin, 2018].

Observe that when the domain of the input noise is restricted to an ℓ_p ball, p^{robust}_σ generalizes the corresponding ℓ_p adversarial robustness. In other words, adversarial robustness is concerned with the quantity $\mathbf{1}(p^{\text{robust}}_\sigma < 1)$, i.e., the indicator function that average-case robustness is less than one (which indicates the presence of an adversarial perturbation), while this work focuses on computing the quantity p^{robust}_σ itself. In the rest of this section, we derive estimators for p^{robust}_σ.

The Monte-Carlo estimator. A naïve estimator of average-case robustness is the Monte-Carlo estimator p^{mc}_σ. It computes the robustness of a classifier f at input $\mathbf{x}$ by generating M noisy samples of $\mathbf{x}$ and then calculating the fraction of these noisy samples that are classified to the same class as $\mathbf{x}$. In other words,

$$\begin{aligned} p^{\text{robust}}_\sigma(\mathbf{x}) &= P_{\epsilon \sim \mathcal{N}(0,\sigma^2)}\left[\arg\max_i f_i(\mathbf{x} + \epsilon) = t\right] \\ &= \mathop{\mathbb{E}}_{\epsilon \sim \mathcal{N}(0,\sigma^2)}\left[\mathbf{1}_{\arg\max_i f_i(\mathbf{x}+\epsilon)=t}\right] \\ &\approx \frac{1}{M}\sum_{j=1}^{M}\left[\mathbf{1}_{\arg\max_i f_i(\mathbf{x}+\epsilon_j)=t}\right] = p^{\text{mc}}_\sigma(\mathbf{x}) \end{aligned}$$

p^{mc}_σ replaces the expectation with the sample average of the M noisy samples of $\mathbf{x}$ and has been used in prior work [Nanda et al., 2021]. Technically, the error for the Monte-Carlo estimator is independent of dimensionality and is given by $\mathcal{O}(1/\sqrt{M})$ [Vershynin, 2018]. However, in practice, for neural networks, p^{mc}_σ requires a large number of random samples to converge to the underlying expectation. For example, for MNIST and CIFAR10 CNNs, it takes around $M = 10,000$ samples per point for p^{mc}_σ to converge, which is computationally expensive, and further, provides little information regarding the decision boundaries of the underlying model. Thus, we set out to address this problem by developing more efficient and informative analytical estimators of average-case robustness.

3.2 ROBUSTNESS ESTIMATION VIA LINEARIZATION

Before deriving analytical robustness estimators for non-linear models, we first consider the simpler problem of deriving this quantity for linear models. This is challenging, especially for multi-class classifiers. For example, given a linear model for a three-class classification problem with weights w_1, w_2, w_3 and biases b_1, b_2, b_3, such that $y = \arg\max_i\{w_i^\top \mathbf{x} + b_i \mid i \in [1, 2, 3]\}$, the decision boundary function between classes i and j is given by $y_{ij} = (w_i - w_j)^\top \mathbf{x} + (b_i - b_j)$. If the predicted label at $\mathbf{x}$ is $y = 1$, the relevant decision boundary functions are y_{12}, y_{13} which characterize the decision boundaries of misclassifications from class 1 to classes $2, 3$ respectively. To compute the total probability of misclassification, we must compute the probability of decision boundaries y_{12}, y_{13} being crossed separately. Crucially, it is important not to "double count" the probability of both y_{12} and y_{13} being simultaneously crossed. Computing the probability of falling into this problematic region is non-trivial, as it depends on the relative orientations of y_{12} and y_{13}. If they are orthogonal, then this problem is avoided, as the probability of crossing y_{12} and y_{13} are independent random variables. However, this is not true in general for non-orthogonal decision boundaries. Further, this "double counting" problem increases in complexity with an increasing number of classes, stemming from a corresponding increase in the number of such pairwise decision

[†] Alternately, if $\epsilon \sim \mathcal{N}(0, \sigma^2/d)$, then $r = \sigma$

boundaries. Lemma 1 provides an elegant solution to this combinatorial problem via the multivariate Gaussian CDF.

Notation: For clarity, we represent tensors by collapsing along the "class" dimension, i.e., $a_i \Big|_{i=1}^{C} := (a_1, a_2, ...a_i, ...a_c)$, where for an order-$t$ tensor a_i, the expansion $a_i \Big|_{i=1}^{C}$ is an order-$(t+1)$ tensor.

Lemma 1. *The local robustness of a multi-class linear model $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$ (with $\mathbf{w} \in \mathbb{R}^{d \times C}$ and $b \in \mathbb{R}^C$) at point $\mathbf{x}$ with respect to a target class t is given by the following. Define weights $\mathbf{u}_i = \mathbf{w}_t - \mathbf{w}_i \in \mathbb{R}^d, \forall i \neq t$, where $\mathbf{w}_t, \mathbf{w}_i$ are rows of $\mathbf{w}$ and biases $c_i = \mathbf{u}_i{}^\top \mathbf{x} + (b_t - b_i) \in \mathbb{R}$. Then,*

$$p_\sigma^{\text{robust}}(\mathbf{x}) = \Phi_{\mathbf{U}\mathbf{U}^\top}\left(\frac{c_i}{\sigma\|\mathbf{u}_i\|_2}\Big|_{\substack{i=1\\ i\neq t}}^{C}\right)$$
$$\text{where } \mathbf{U} = \frac{\mathbf{u}_i}{\|\mathbf{u}_i\|_2}\Big|_{\substack{i=1\\ i\neq t}}^{C} \in \mathbb{R}^{(C-1)\times d}$$

and $\Phi_{\mathbf{U}\mathbf{U}^\top}$ is the $(C-1)$-dimensional Normal CDF with zero mean and covariance $\mathbf{U}\mathbf{U}^\top$.

Proof Idea. The proof involves constructing decision boundary functions $g_i(\mathbf{x}) = f_t(\mathbf{x}) - f_i(\mathbf{x})$ and computing the probability $p_\sigma^{\text{robust}}(\mathbf{x}) = P_\epsilon(\bigcup_{\substack{i=1\\ i\neq t}}^{C} g_i(\mathbf{x}+\epsilon) > 0)$. For Gaussian ϵ, we observe that $\frac{\mathbf{u}}{\sigma\|\mathbf{u}\|_2}{}^\top \epsilon \sim \mathcal{N}(0,1)$ is also a Gaussian, which applied vectorially results in our usage of Φ. As convention, we represent $\mathbf{U}$ in a normalized form to ensure that its rows are unit norm. □

The proof is in Appendix A.1. Thus, the multivariate Gaussian CDF provides an elegant solution to the previously mentioned "double counting" problem. Here, the matrix $\mathbf{U}$ exactly captures the linear decision boundaries, and the covariance matrix $\mathbf{U}\mathbf{U}^\top$ encodes the alignment between pairs of decision boundaries of different classes.

Remark. For the binary classification case, we get $\mathbf{U}\mathbf{U}^\top = 1$ (a scalar), and $p_\sigma^{\text{robust}}(\mathbf{x}) = \phi(\frac{c}{\sigma\|\mathbf{u}\|_2})$, where ϕ is the CDF of the scalar standard normal, which was previously also shown by Weng et al. [2019], Pawelczyk et al. [2023]. Hence Lemma 1 is a multi-class generalization of these works.

If the decision boundary vectors $\mathbf{u}_i$ are all orthogonal to each other, then the covariance matrix $\mathbf{U}\mathbf{U}^\top$ is the identity matrix. For diagonal covariance matrices, the multivariate Normal CDF (*mvn-cdf*) can be written as the product of univariate Normal CDFs, which is easy to compute. However, in practice, we find that the covariance matrix is strongly non-diagonal, indicating that the decision boundaries are not orthogonal to each other. This non-diagonal nature of covariance matrices in practice leads to the resulting *mvn-cdf* not having a closed form solution, and thus needing to be approximated via sampling [Botev, 2017, Sci]. However, this sampling is performed in the $(C-1)$-dimensional space as opposed to the d-dimensional space that p_σ^{mc} samples from. In practice, for classification problems, we often have $C << d$, making sampling in $(C-1)$-dimensions more efficient. We would like to stress here that the expression in Lemma 1 represents the simplest expression to compute the average-case robustness: the usage of the multi-variate Gaussian CDF cannot be avoided due to the computational nature of this problem. We now discuss the applicability of Lemma 1 to non-Gaussian noise.

Lemma 2. ***(Application to non-Gaussian noise)*** *For high-dimensional data ($d \to \infty$), Lemma 1 generalizes to any coordinate-wise independent noise distribution that satisfies Lyapunov's condition.*

Proof Idea. Applying Lyupanov's central limit theorem [Patrick, 1995], given $\epsilon \sim \mathcal{R}$ is sampled from some distribution $\mathcal{R}$, we have $\frac{\mathbf{u}}{\sigma\|\mathbf{u}\|_2}{}^\top \epsilon = \sum_{j=1}^{d} \frac{\mathbf{u}_j}{\sigma\|\mathbf{u}\|_2}\epsilon_j \xrightarrow{d} \mathcal{N}(0,1)$, which holds as long as the sequence $\{\frac{\mathbf{u}_j}{\|\mathbf{u}\|_2}\epsilon_j\}$ are independent random variables and satisfy the Lyapunov condition, which encodes the fact that higher-order moments of such distributions progressively shrink. □

Thus, as long as the input noise distribution is "well-behaved", the central limit theorem ensures that the distribution of high-dimensional dot products is Gaussian, thus motivating our use of the *mvn-cdf* more generally beyond Gaussian input perturbations. We note that it is also possible to easily generalize Lemma 1 to **non-isotropic** Gaussian perturbations with a covariance matrix $\mathcal{C}$, which only changes the form of the covariance matrix of the *mvn-cdf* from $\mathbf{U}\mathbf{U}^\top \to \mathbf{U}\mathcal{C}\mathbf{U}^\top$, which we elaborate in Appendix A.1. In the rest of this paper, we focus on the isotropic case.

3.2.1 Estimator 1: The Taylor Estimator

Using the estimator derived for multi-class linear models in Lemma 1, we now derive the Taylor estimator, a local robustness estimator for non-linear models.

Definition 2. *The **Taylor estimator** for the local robustness of a classifier f at point $\mathbf{x}$ with respect to target class t is given by linearizing f around $\mathbf{x}$ using a first-order Taylor expansion, with decision boundaries $g_i(\mathbf{x}) = f_t(\mathbf{x}) - f_i(\mathbf{x})$, $\forall i \neq t$, leading to*

$$p_\sigma^{\text{taylor}}(\mathbf{x}) = \Phi_{\mathbf{U}\mathbf{U}^\top}\left(\frac{g_i(\mathbf{x})}{\sigma\|\nabla_{\mathbf{x}} g_i(\mathbf{x})\|_2}\Big|_{\substack{i=1\\ i\neq t}}^{C}\right)$$

with $\mathbf{U}$ and Φ defined as in the linear case.

The proof is in Appendix A.1. It involves locally linearizing non-linear decision boundary functions $g_i(\mathbf{x})$ using a Taylor series expansion. We expect this estimator to have a small

error when the underlying model is well-approximated by a locally linear function in the local neighborhood. We formalize this intuition by computing the estimation error for a quadratic classifier.

Proposition 1. *The **estimation error** of the Taylor estimator for a classifier with a quadratic decision boundary $g_i(\mathbf{x}) = \mathbf{x}^\top A_i \mathbf{x} + \mathbf{u}_i^\top \mathbf{x} + c_i$ and positive-semidefinite A_i is upper bounded by*

$$|p_\sigma^{\text{robust}}(\mathbf{x}) - p_\sigma^{\text{taylor}}(\mathbf{x})| \leq k\sigma^{C-1} \prod_{\substack{i=1 \\ i \neq t}}^{C} \frac{\lambda_{\max}^{A_i}}{\|\mathbf{u}_i\|_2}$$

for noise $\epsilon \sim \mathcal{N}(0, \sigma^2/d)$, in the limit of $d \to \infty$. Here, $\lambda_{\max}^{A_i}$ is the max eigenvalue of A_i, and k is a small problem dependent constant.

The proof is in Appendix A.1. This statement formalizes two key intuitions with regards to the Taylor estimator: (1) the estimation error depends on the size of the local neighborhood σ (the smaller the local neighborhood, the more locally linear the model, and the smaller the estimator error), and (2) the estimation error depends on the extent of non-linearity of the underlying function, which is given by the ratio of the max eigenvalue of A to the Frobenius norm of the linear term. This measure of non-linearity of a function, called normalized curvature, has also been independently proposed by previous work [Srinivas et al., 2022]. Notably, if the max eigenvalue is zero, then the function $g_i(\mathbf{x})$ is exactly linear, and the estimation error is zero, reverting back to the linear case in Lemma 1.

3.2.2 Estimator 2: The MMSE Estimator

While the Taylor estimator is more efficient than the Monte Carlo estimator, it has a drawback: its linearization is only faithful at perturbations close to the data point and not necessarily for larger perturbations. To mitigate this issue, we use a form of linearization that is faithful over larger noise perturbations. Linearization has been studied in feature attribution research, which concerns itself with approximating non-linear models with linear ones to produce model explanations [Han et al., 2022]. In particular, the SmoothGrad [Smilkov et al., 2017] technique has been described as the MMSE (minimum mean-squared error) optimal linearization of the model [Han et al., 2022, Agarwal et al., 2021] in a Gaussian neighborhood around the data point. Using a similar idea, we propose the MMSE estimator p_σ^{mmse} as follows.

Definition 3. *The **MMSE estimator** for the local robustness of a classifier f at point $\mathbf{x}$ with respect to target class t is given by an MMSE linearization f around $\mathbf{x}$, for decision boundaries $g_i(\mathbf{x}) = f_t(\mathbf{x}) - f_i(\mathbf{x}), \forall i \neq t$, leading to*

$$p_\sigma^{\text{mmse}}(\mathbf{x}) = \Phi_{\mathbf{U}\mathbf{U}^\top} \left(\frac{\tilde{g}_i(\mathbf{x})}{\sigma \|\nabla_{\mathbf{x}} \tilde{g}_i(\mathbf{x})\|_2} \Big|_{\substack{i=1 \\ i \neq t}}^{C} \right)$$

$$\text{where } \tilde{g}_i(\mathbf{x}) = \frac{1}{N} \sum_{j=1}^{N} g_i(\mathbf{x} + \epsilon) \ , \ \epsilon \sim \mathcal{N}(0, \sigma^2)$$

with $\mathbf{U}$ and Φ defined as in the linear case, and N is the number of perturbations.

The proof is in Appendix A.1. It involves creating a randomized smooth model [Cohen et al., 2019] from the base model and computing the decision boundaries of this smooth model. Note that this estimator also involves drawing noise samples like the Monte Carlo estimator. However, unlike the Monte Carlo estimator, we find that the MMSE estimator converges fast (around $N = 5$), leading to an empirical advantage. We now compute the estimation error of the MMSE estimator.

Proposition 2. *The **estimation error** of the MMSE estimator for a classifier with a quadratic decision boundary $g_i(\mathbf{x}) = \mathbf{x}^\top A_i \mathbf{x} + \mathbf{u}_i^\top \mathbf{x} + c_i$ and positive-semidefinite A_i is upper bounded by*

$$|p_\sigma^{\text{robust}}(\mathbf{x}) - p_\sigma^{\text{mmse}}(\mathbf{x})| \leq k\sigma^{C-1} \prod_{\substack{i=1 \\ i \neq t}}^{C} \frac{\lambda_{\max}^{A_i} - \lambda_{\textit{mean}}^{A_i}}{\|\mathbf{u}_i\|_2}$$

for noise $\epsilon \sim \mathcal{N}(0, \sigma^2/d)$, in the limit of $d \to \infty$ and $N \to \infty$. Here, $\lambda_{\max}^{A_i}, \lambda_{\textit{mean}}^{A_i}$ are the maximum and mean eigenvalue of A_i respectively, and k is a small problem dependent constant.

The proof is in Appendix A.1. The result above highlights two aspects of the MMSE estimator: (1) it incurs a smaller estimation error than the Taylor estimator, and (2) even in the limit of large number of samples $N \to \infty$, the error of the MMSE estimator is non-zero, except when $\lambda_{\text{mean}}^{A_i} = \lambda_{\max}^{A_i}$. For PSD matrices, this becomes zero when A_i is a multiple of the identity matrix [‡], reverting back to the linear case in Lemma 1.

3.2.3 (Optionally) Approximating *mvn-cdf*: Connecting Robustness Estimation with Softmax

Approximation with Multivariate Sigmoid. One drawback of the Taylor and MMSE estimators is their use of the *mvn-cdf*, which does not have a closed form solution and can cause the estimators to be slow for settings with a large number of classes C. In addition, the *mvn-cdf* makes

[‡]When $d \to \infty$, $\epsilon^\top A \epsilon = \lambda \|\epsilon\|^2 = \lambda \sigma^2$ is a constant, and thus an isotropic quadratic function resembles a linear one in this neighborhood.

these estimators non-differentiable, which is inconvenient for applications which require differentiating p_σ^{robust}. To alleviate these issues, we approximate the *mvn-cdf* with an analytical closed-form expression. As CDFs are monotonically increasing functions, the approximation should also be monotonically increasing.

To this end, it has been previously shown that the *univariate* Normal CDF ϕ is well-approximated by the sigmoid function [Hendrycks and Gimpel, 2016]. It is also known that when $\mathbf{U}\mathbf{U}^\top = I$, *mvn-cdf* is given by $\Phi(\mathbf{x}) = \prod_i \phi(\mathbf{x}_i)$, i.e., it is given by the product of the univariate normal CDFs. Thus, we may choose to approximate $\Phi(\mathbf{x}) = \prod_i \text{sigmoid}(\mathbf{x})$. However, when the inputs are small, this can be simplified as follows:

$$\begin{aligned}\Phi_I(\mathbf{x}) &= \prod_i \phi(\mathbf{x}_i) \approx \prod_i \frac{1}{1+\exp(-\mathbf{x}_i)} \\ &= \frac{1}{1+\sum_i \exp(-\mathbf{x}_i) + \sum_{j,k} \exp(-\mathbf{x}_j - \mathbf{x}_k) + ...} \\ &\approx \frac{1}{1+\sum_i \exp(-\mathbf{x}_i)} \quad (\text{for } \mathbf{x}_i \to \infty \ \forall i)\end{aligned}$$

We call the final expression the "multivariate sigmoid" (*mv-sigmoid*) which serves as our approximation of *mvn-cdf*, especially at the tails of the distribution. While we expect estimators using *mv-sigmoid* to approximate ones using *mvn-cdf* only when $\mathbf{U}\mathbf{U}^\top = \mathbf{I}$, we find experimentally that the approximation works well even for practical values of the covariance matrix $\mathbf{U}\mathbf{U}^\top$. Using this approximation to substitute *mv-sigmoid* for *mvn-cdf* in the p_σ^{taylor} and p_σ^{mmse} estimators yields the $p_\sigma^{\text{taylor\_mvs}}$ and $p_\sigma^{\text{mmse\_mvs}}$ estimators, respectively. We present further analysis on the multivariate sigmoid in Appendix A.4.

Approximation with Softmax. A common method to estimate the confidence of model predictions is to use the softmax function applied to the logits $f_i(\mathbf{x})$ of a model. We note that softmax is identical to *mv-sigmoid* when directly applied to the logits of neural networks:

$$\text{softmax}_t\left(f_i(\mathbf{x})\,\middle|_{i=1}^{C}\right) = \frac{\exp(f_t(\mathbf{x}))}{\sum_{i=1}^{C}\exp(f_i(\mathbf{x}))} = \frac{1}{1+\sum\limits_{\substack{i=1\\ i\neq t}}^{C}\exp(f_i(\mathbf{x})-f_t(\mathbf{x}))} = \text{mv-sigmoid}\left(g_i(\mathbf{x})\,\middle|_{\substack{i=1\\ i\neq t}}^{C}\right)$$

Recall that $g_i(\mathbf{x}) = f_t(\mathbf{x}) - f_i(\mathbf{x})$ is the decision boundary function. Note that this equivalence only holds for the specific case of logits, and cannot be applied to approximate the Taylor estimator, for instance. Nonetheless, given this similarity, it is reasonable to ask whether softmax applied to logits (henceforth p_T^{softmax} for softmax with temperature T) itself can be a "good enough" estimator of p_σ^{robust} in practice. In other words, does p_T^{softmax} well-approximate p_σ^{robust} in certain settings? In Appendix A.1, we provide a theoretical result for a restricted linear setting where softmax can indeed match the behavior of $p_\sigma^{\text{taylor\_mvs}}$, which happens precisely when $\mathbf{U}\mathbf{U}^\top = \mathbf{I}$ and all the class-wise gradients are equal. In the next section, we demonstrate empirically that the softmax estimator p_T^{softmax} is a poor estimator of average-case robustness in practice.

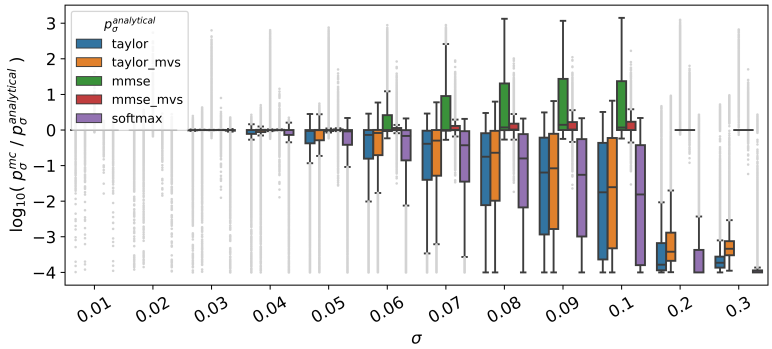
4 EMPIRICAL EVALUATION

In this section, we first evaluate the estimation errors and computational efficiency of the analytical estimators, and then evaluate the impact of robustness training within models on these estimation errors. Then, we analyze the relationship between average-case robustness and softmax probability. Lastly, we demonstrate the usefulness of local robustness for model and dataset understanding with two case studies. Key results are discussed in this section and full results are in Appendix A.4.

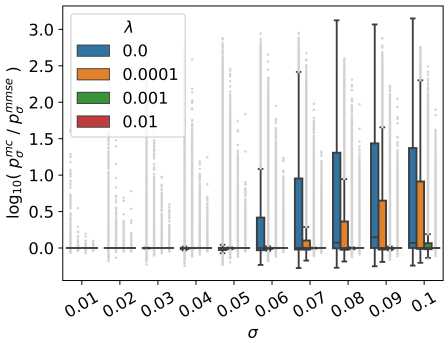
Datasets and models. We evaluate the estimators on four datasets: MNIST [Deng, 2012], FashionMNIST [Xiao et al., 2017], CIFAR10 [Krizhevsky et al., 2009], and CIFAR100 [Krizhevsky et al., 2009]. For MNIST and FashionMNIST, we train linear models and CNNs. For CIFAR10 and CIFAR100, we train Transformer models. We also train ResNet18 models [He et al., 2016] using varying levels of gradient norm regularization [Srinivas and Fleuret, 2018, Srinivas et al., 2024] to obtain models with varying levels of robustness. For gradient norm regularization, the objective function is $\ell(f(x), y) + \lambda\|\nabla_x f(x)\|_2^2$, where λ is the regularization constant. The larger λ is, the more robust the model. Note that gradient norm regularization is equivalent to Gaussian data augmentation with an infinite number of augmented samples [Srinivas and Fleuret, 2018] and is different from adversarial training. Unless otherwise noted, the experiments below use each dataset's test set which consists of 10,000 points. Additional details about the datasets and models are described in Appendix A.2 and A.3.

4.1 EVALUATION OF THE ESTIMATION ERRORS OF ANALYTICAL ESTIMATORS

The analytical estimators accurately compute local robustness. To empirically evaluate the estimation error of our estimators, we calculate p_σ^{robust} for each model using p_σ^{mc}, p_σ^{taylor}, p_σ^{mmse}, $p_\sigma^{\text{taylor\_mvs}}$, $p_\sigma^{\text{mmse\_mvs}}$, and p_T^{softmax} for different σ values. For p_σ^{mc}, p_σ^{mmse}, and $p_\sigma^{\text{mmse\_mvs}}$, we use a sample size at which these estimators have converged ($n = 10000, 500,$ and 500, respectively). (Convergence analyses are in Appendix A.4.) We take the Monte-



(a) CIFAR10, ResNet18
Comparing estimator errors



(b) CIFAR10, ResNet18
Varying model robustness

Figure 2: Empirical evaluation of analytical estimators. (a) The smaller the noise neighborhood σ, the more accurately the estimators compute p_σ^{robust}. p_σ^{mmse} and $p_\sigma^{\text{mmse\_mvs}}$ are the best estimators of p_σ^{robust}, followed closely by $p_\sigma^{\text{taylor\_mvs}}$ and p_σ^{taylor}, trailed by p_T^{softmax}. (b) For more robust models, the estimators compute p_σ^{robust} more accurately over a larger σ. Together, these results indicate that the analytical estimators accurately compute p_σ^{robust}.

Carlo estimator as the gold standard estimate of p_σ^{robust}), and compute the absolute and relative difference between p_σ^{mc} and the other estimators to evaluate their estimation errors.

The performance of the estimators for the CIFAR10 ResNet18 model is shown in Figure 2a. The results indicate that $p_\sigma^{\text{mmse\_mvs}}$ and p_σ^{mmse} are the best estimators of p_σ^{robust}, followed closely by $p_\sigma^{\text{taylor\_mvs}}$ and p_σ^{taylor}, trailed by p_T^{softmax}. This is consistent with the theory in Section 3, where the analytical estimation errors of p_σ^{mmse} are lower than p_σ^{taylor}.

The results also confirm that the smaller the noise neighborhood σ, the more accurately the estimators compute p_σ^{robust}. For the MMSE and Taylor estimators, this is because their linear approximation of the model around the input is more faithful for smaller σ. As expected, when the model is linear, p_σ^{taylor} and p_σ^{mmse} accurately compute p_σ^{robust} for all σ's (Appendix A.4). For the softmax estimator, p_T^{softmax} values are constant over σ and this particular model has high p_T^{softmax} values for most points. Thus, for small σ's where p_σ^{robust} is near one, p_T^{softmax} happens to approximate p_σ^{robust} for this model. Examples of images with varying levels of noise (σ) are in Appendix A.4.

Impact of robust training on estimation errors. The performance of p_σ^{mmse} for CIFAR10 ResNet18 models of varying levels of robustness is shown in Figure 2b. The results indicate that the estimator is more accurate for more robust models (larger λ) over a larger σ. This is because robust training leads to models that are more locally linear [Moosavi-Dezfooli et al., 2019], making the estimator's linear approximation of the model around the input more accurate over a larger σ, making its p_σ^{robust} values more accurate.

Evaluating estimation error of mv-sigmoid. To examine *mv-sigmoid*'s approximation of *mvn-cdf*, we compute both functions using the same inputs ($z = \frac{g_i(\mathbf{x})}{\sigma \|\nabla_{\mathbf{x}} g_i(\mathbf{x})\|_2} |_{i \neq t}^{C}$, as described in Proposition 2) for the CIFAR10 ResNet18 model for different σ. The plot of *mv-sigmoid(z)* against *mvn-cdf(z)* for $\sigma = 0.05$ is shown in Appendix A.4 (Figure 10). The results indicate that the two functions are strongly positively correlated with low approximation error, suggesting that *mv-sigmoid* approximates the *mvn-cdf* well in practice.

4.2 EVALUATION OF COMPUTATIONAL EFFICIENCY OF ANALYTICAL ESTIMATORS

The analytical estimators are more efficient than the naïve estimator. We examine the efficiency of the estimators by measuring their runtimes when calculating $p_{\sigma=0.1}^{\text{robust}}$ for the CIFAR10 ResNet18 model for 50 points. Runtimes are displayed in Table 1. They indicate that p_σ^{taylor} and p_σ^{mmse} perform 35x and 17x faster than p_σ^{mc}, respectively. Additional runtimes are in Appendix A.4.

We also examine the efficiency of the analytical estimators in terms of memory usage. The backward pass is observed to take about twice the amount of floating-point operations (FLOPs) as a forward pass [flo]. In addition, we performed an experiment and found that a forward and backward pass uses about twice the peak memory of a single forward pass. Thus, each iteration of p_σ^{mmse} (which consists of a forward and backward pass) is roughly 3x the number of FLOPs and twice the peak memory of a single iteration of p_σ^{mc} (which consists of one forward pass). However, p_σ^{mmse} requires 5 iterations for convergence while p_σ^{mc} requires about 10,000. Thus, overall, p_σ^{mmse} is more memory-efficient than p_σ^{mc}.

Estimator	Number of Samples (n)	CPU Runtime (h:m:s)	GPU Runtime (h:m:s)
p_σ^{mc}	$n = 10,000$	1:41:11	0:19:56
p_σ^{taylor}	N/A	0:00:08	0:00:02
p_σ^{mmse}	$n = 5$	0:00:41	0:00:06

Table 1: Runtimes of p_σ^{robust} estimators. Each estimator computes $p_{\sigma=0.1}^{\text{robust}}$ for the CIFAR10 ResNet18 model for 50 data points. Estimators that use sampling use the minimum number of samples necessary for convergence. Runtimes are in the format of hour:minute:second. The GPU used was a Tesla V100. The analytical estimators (p_σ^{taylor} and p_σ^{mmse}) are more efficient than the naïve estimator (p_σ^{mc}).

4.3 CASE STUDIES

Identifying non-robust data points. While robustness is typically viewed as the property of a model, the average-case robustness perspective compels us to view robustness as a joint property of both the model and the data point. In light of this, we can ask, given the same model, which samples are robustly and non-robustly classified? We evaluate whether p_σ^{robust} can distinguish such images better than p_T^{softmax}. To this end, we train a simple CNN to distinguish between images with high and low p_σ^{mmse} and the same CNN to also distinguish between images with high and low p_T^{softmax} (additional setup details described in Appendix A.4). Then, we compare the performance of the two models. For CIFAR10, the test set accuracy for the p_σ^{mmse} CNN is $\mathbf{92\%}$ while that for the p_T^{softmax} CNN is $\mathbf{58\%}$. These results indicate that p_σ^{robust} better identifies images that are robust to and vulnerable to random noise than p_T^{softmax}.

We also present visualizations of images with the highest and lowest p_σ^{mmse} in each class for each model. For comparison, we do the same with p_T^{softmax}. Example CIFAR10 images are shown in Figure 3. We observe that images with low p_σ^{robust} tend to have neutral colors, with the object being a similar color as the background (making the prediction likely to change when the image is slightly perturbed), while images with high p_σ^{robust} tend to be brightly-colored, with the object strongly contrasting with the background (making the prediction likely to stay constant when the image is slightly perturbed). Recall that points with small p_σ^{robust} are close to the decision boundary, while those farther away have a high p_σ^{robust}. Thus, high p_σ^{robust} points may be thought of as "canonical" examples of the underlying class, while low p_σ^{robust} examples are analogous to "support vectors", that are critical to model learning. These results showcase the utility of average-case robustness for dataset exploration and analysis, particularly in identifying canonical and outlier examples.

Detecting robustness bias among classes: Is the model differently robust for different classes? We also demonstrate that p_σ^{robust} can detect bias in local robustness [Nanda et al., 2021] by examining its distribution for each class for each model and test set over different σ's. Results for the CIFAR10 ResNet18 model are in plotted in Figure 4. The results show that different classes have significantly different p_σ^{robust} distributions, i.e., the model is significantly more robust for some classes (e.g., frog) than for others (e.g., airplane). Similarly for the FMNIST CNN case in Figure 4, we find that the pullover class is much less robust than the sandal class. This observation indicates a disparity in outcomes for these different classes, and underscores the importance of evaluating per-class and per-datapoint robustness metrics before deploying models in the wild. The results also show that p_σ^{mc} and p_σ^{mmse} have very similar distributions, further indicating that the latter well-approximates the former. p_σ^{robust} detects robustness bias across all other models and datasets too: MNIST CNN, and CIFAR100 ResNet18 (Appendix A.4). Thus, p_σ^{robust} can be applied to detect robustness bias among classes, which is critical when models are deployed in high-stakes, real-world settings.

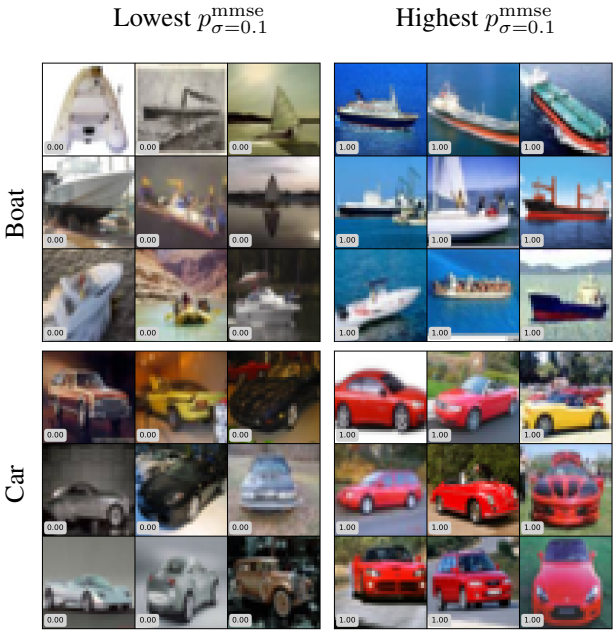
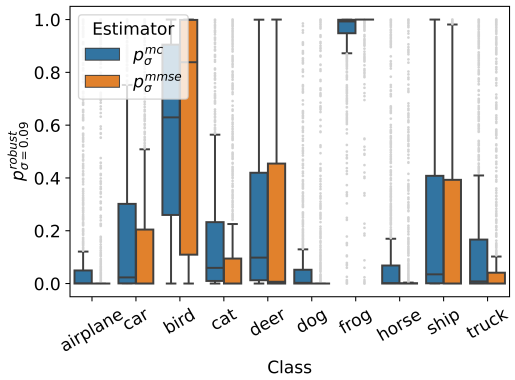


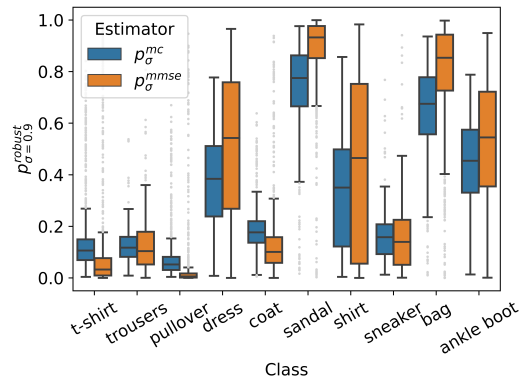
Figure 3: Example ranking of p_σ^{robust} among CIFAR10 classes. Images with high p_σ^{robust} are farther away from the decision boundary, and tend to be brighter and have stronger object-background contrast than those with low p_σ^{robust}, which are closer to the decision boundary, and thus easily misclassified.

5 CONCLUSION

In this work, we find that adversarial robustness does not provide a comprehensive picture of model behavior, and to this end, we propose the usage of average-case robustness. While adversarial robustness is more suited for applica-



(a) CIFAR10, ResNet18



(b) FMNIST, CNN

Figure 4: Computing robustness bias among classes for the (a) ResNet18 CIFAR10 model, and (b) for the CNN FMNIST model. p_σ^{robust} reveals that the model robustness varies significantly across classes, revealing a marked class-wise bias within standard models. The analytical estimator p_σ^{mmse} accurately captures this model bias.

tions in model security, average-case robustness is suited for model understanding and debugging. To our knowledge, this work is the first to investigate analytical estimators for average-case robustness in a multi-class setting. The analytical aspect of these estimators helps understand average-case robustness via model decision boundaries, and also connect to ideas such as Randomized Smoothing (via the MMSE estimator) and softmax probabilities.

Future research directions include exploring additional applications of average-case robustness, such as training average-case robust models that minimize the probability of misclassification and debugging-oriented applications such as detecting model memorization, and dataset outliers.

Code availability. Code is available at `https://github.com/AI4LIFE-GROUP/average-case-robustness`.

References

SciPy multivariate normal CDF. `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.multivariate_normal.html`.

What's the backward-forward FLOP ratio for neural networks? `https://www.lesswrong.com/posts/fnjKpBoWJXcSDwhZk/what-s-the-backward-forward-flop-ratio-for-neu` Accessed: 2024-04-04.

Sushant Agarwal, Shahin Jabbari, Chirag Agarwal, Sohini Upadhyay, Steven Wu, and Himabindu Lakkaraju. Towards the unification and robustness of perturbation and gradient based explanations. *International Conference on Machine Learning*, 2021.

Zdravko I Botev. The normal law under linear restrictions: Simulation and estimation via minimax tilting. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 2017.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. *IEEE Symposium on Security and Privacy*, 2017.

Nicholas Carlini, Florian Tramer, Krishnamurthy Dvijotham, Leslie Rice, Mingjie Sun, and Zico Kolter. (Certified!!) adversarial robustness for free! *International Conference on Learning Representations*, 2022.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. *International Conference on Machine Learning*, 2019.

Li Deng. The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 2012.

Mahyar Fazlyab, Manfred Morari, and George J Pappas. Probabilistic verification and reachability analysis of neural networks via semidefinite programming. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 2726–2731. IEEE, 2019.

Jean-Yves Franceschi, Alhussein Fawzi, and Omar Fawzi. Robustness of classifiers to uniform lp and gaussian noise. In *International Conference on Artificial Intelligence and Statistics*, pages 1280–1288. PMLR, 2018.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *International Conference on Learning Representations*, 2015.

Tessa Han, Suraj Srinivas, and Himabindu Lakkaraju. Which explanation should I choose? A function approximation perspective to characterizing post hoc explanations. *Advances in Neural Information Processing Systems*, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *University of Toronto*, 2009.

Aounon Kumar, Alexander Levine, Soheil Feizi, and Tom Goldstein. Certifying confidence via randomized smoothing. *Advances in Neural Information Processing Systems*, 33:5165–5177, 2020.

Ravi Mangal, Aditya V Nori, and Alessandro Orso. Robustness of neural networks: A probabilistic and practical approach. In *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 93–96. IEEE, 2019.

Mazda Moayeri, Kiarash Banihashem, and Soheil Feizi. Explicit tradeoffs between adversarial and natural distributional robustness. *Advances in Neural Information Processing Systems*, 2022.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. DeepFool: A simple and accurate method to fool deep neural networks. *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9078–9086, 2019.

Vedant Nanda, Samuel Dooley, Sahil Singla, Soheil Feizi, and John P Dickerson. Fairness through robustness: Investigating robustness disparity in deep learning. *ACM Conference on Fairness, Accountability, and Transparency*, 2021.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift. *Advances in Neural Information Processing Systems*, 2019.

Billingsley Patrick. Probability and measure. *A Wiley-Interscience Publication, John Wiley*, 118:119, 1995.

Martin Pawelczyk, Teresa Datta, Johannes van-den Heuvel, Gjergji Kasneci, and Himabindu Lakkaraju. Probabilistically robust recourse: Navigating the trade-offs between costs and robustness in algorithmic recourse. *International Conference on Learning Representations*, 2023.

Leslie Rice, Anna Bair, Huan Zhang, and J Zico Kolter. Robustness between the worst and average case. *Advances in Neural Information Processing Systems*, 34: 27840–27851, 2021.

Alexander Robey, Luiz Chamon, George J Pappas, and Hamed Hassani. Probabilistically robust learning: Balancing average and worst-case performance. In *International Conference on Machine Learning*, pages 18667–18686. PMLR, 2022.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. SmoothGrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

Suraj Srinivas and François Fleuret. Knowledge transfer with jacobian matching. *International Conference on Machine Learning*, 2018.

Suraj Srinivas, Kyle Matoba, Himabindu Lakkaraju, and François Fleuret. Efficient training of low-curvature neural networks. *Advances in Neural Information Processing Systems*, 35:25951–25964, 2022.

Suraj Srinivas, Sebastian Bordt, and Himabindu Lakkaraju. Which models have perceptually-aligned gradients? An explanation via off-manifold robustness. *Advances in Neural Information Processing Systems*, 2024.

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. *Advances in Neural Information Processing Systems*, 2020.

Sunil Thulasidasan, Sushil Thapa, Sayera Dhaubhadel, Gopinath Chennupati, Tanmoy Bhattacharya, and Jeff Bilmes. An effective baseline for robustness to distributional shift. *IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2021.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*. Cambridge University Press, 2018.

Lily Weng, Pin-Yu Chen, Lam Nguyen, Mark Squillante, Akhilan Boopathy, Ivan Oseledets, and Luca Daniel. Proven: Verifying robustness of neural networks with a probabilistic approach. In *International Conference on Machine Learning*, pages 6727–6736. PMLR, 2019.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

A APPENDIX

A.1 PROOFS

Lemma A.1. *The local robustness of a multi-class linear model $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$ (with $\mathbf{w} \in \mathbb{R}^{d \times C}$ and $b \in \mathbb{R}^C$) at point $\mathbf{x}$ with respect to a target class t is given by the following. Define weights $\mathbf{u}_i = \mathbf{w}_t - \mathbf{w}_i \in \mathbb{R}^d, \forall i \neq t$, where $\mathbf{w}_t, \mathbf{w}_i$ are rows of $\mathbf{w}$ and biases $c_i = {\mathbf{u}_i}^\top \mathbf{x} + (b_t - b_i) \in \mathbb{R}$. Then,*

$$p_\sigma^{robust}(\mathbf{x}) = \Phi_{\mathbf{U}\mathbf{U}^\top} \left(\frac{c_i}{\sigma \|\mathbf{u}_i\|_2} \Big|_{\substack{i=1 \\ i \neq t}}^{C} \right)$$
$$\text{where } \mathbf{U} = \frac{\mathbf{u}_i}{\|\mathbf{u}_i\|_2} \Big|_{\substack{i=1 \\ i \neq t}}^{C} \in \mathbb{R}^{(C-1) \times d}$$

and $\Phi_{\mathbf{U}\mathbf{U}^\top}$ is the $(C-1)$-dimensional Normal CDF with zero mean and covariance $\mathbf{U}\mathbf{U}^\top$.

Proof. First, we rewrite p_σ^{robust} in the following manner, by defining $g_i(\mathbf{x}) = f_t(\mathbf{x}) - f_i(\mathbf{x}) > 0$, which is the "decision boundary function".

$$\begin{aligned} p_\sigma^{\text{robust}} &= P_{\epsilon \sim \mathcal{N}(0,\sigma^2)} \left[\max_i f_i(\mathbf{x} + \epsilon) < f_t(\mathbf{x} + \epsilon) \right] \\ &= P_{\epsilon \sim \mathcal{N}(0,\sigma^2)} \left[\bigcup_{i=1; i \neq t}^{C} g_i(\mathbf{x} + \epsilon) > 0 \right] \end{aligned}$$

Now, assuming that f, g are linear such that $g_i(\mathbf{x}) = {\mathbf{u}_i'}^\top \mathbf{x} + g(0)$, we have $g_i(\mathbf{x} + \epsilon) = g_i(\mathbf{x}) + {\mathbf{u}_i}^\top \epsilon$, and obtain

$$p_\sigma^{\text{robust}} = P_{\epsilon \sim \mathcal{N}(0,\sigma^2)} \left[\bigcup_{i=1; i \neq t}^{C} {\mathbf{u}_i}^\top \epsilon > -g_i(\mathbf{x}) \right] \tag{1}$$
$$= P_{z \sim \mathcal{N}(0,I_d)} \left[\bigcup_{i=1; i \neq t}^{C} \frac{\mathbf{u}_i}{\|\mathbf{u}_i\|_2}^\top z > -\frac{g_i(\mathbf{x})}{\sigma \|\mathbf{u}_i\|_2} \right] \tag{2}$$

This step simply involves rescaling and standardizing the Gaussian to be unit normal. We now make the following observations:

- For any matrix $\mathbf{U} \in \mathbb{R}^{C-1 \times d}$ and a d-dimensional Gaussian random variable $z \sim \mathcal{N}(0, I_d) \in \mathbb{R}^d$, we have $\mathbf{U}^\top z \sim \mathcal{N}(0, \mathbf{U}\mathbf{U}^\top)$, i.e., an (C-1) -dimensional Gaussian random variable.
- CDF of a multivariate Gaussian RV is defined as $P_z[\bigcup_i z_i < t_i]$ for some input values t_i

Using these observations, if we construct $\mathbf{U} = \frac{\mathbf{u}_i}{\|\mathbf{u}_i\|_2} \Big|_{\substack{i=1 \\ i \neq t}}^{C} \in \mathbb{R}^{(C-1) \times d}$, and obtain

$$\begin{aligned} p_\sigma^{\text{robust}} &= P_{r \sim \mathcal{N}(0, \mathbf{U}\mathbf{U}^\top)} \left[\bigcup_{i=1; i \neq t}^{C} r_i < \frac{g_i(\mathbf{x})}{\sigma \|\mathbf{u}_i\|_2} \right] \\ &= \text{CDF}_{\mathcal{N}(0, UU^\top)} \left(\frac{g_i(\mathbf{x})}{\sigma \|\mathbf{u}_i\|_2} \Big|_{\substack{i=1 \\ i \neq t}}^{C} \right) \end{aligned}$$

where $g_i(\mathbf{x}) = \mathbf{u}_i{}^\top \mathbf{x} + g_i(0) = (\mathbf{w}_t - \mathbf{w}_i)^\top \mathbf{x} + (b_t - b_i)$

□

Lemma A.2. ***(Extension to non-Gaussian noise)*** *For high-dimensional data ($d \to \infty$), Lemma 1 generalizes to any coordinate-wise independent noise distribution that satisfies Lyapunov's condition.*

Proof. Applying Lyupanov's central limit theorem, given $\epsilon \sim \mathcal{R}$ is sampled from some distribution $\mathcal{R}$ to equation 2 in the previous proof, we have we have $\frac{\mathbf{u}}{\sigma\|\mathbf{u}\|_2}{}^\top \epsilon = \sum_{j=1}^{d} \frac{\mathbf{u}_j}{\sigma\|\mathbf{u}\|_2}\epsilon_j \xrightarrow{d} \mathcal{N}(0,1)$, which holds as long as the sequence $\{\frac{\mathbf{u}_j}{\|\mathbf{u}\|_2}\epsilon_j\}$ are independent random variables and satisfy the Lyapunov condition. In particular, this implies that $\mathbf{U}^\top z \sim \mathcal{N}(0, \mathbf{U}\mathbf{U}^\top)$, and the proof proceeds as similar to the Gaussian case after this step. □

Lemma A.3. ***(Extension to non-isotropic Gaussian)*** *Lemma 1 can be extended to the case of $\epsilon \sim \mathcal{N}(0, \mathcal{C})$ for an arbitrary positive definite covariance matrix $\mathcal{C}$:*

$$p_\sigma^{robust}(\mathbf{x}) = \Phi_{\mathbf{U}\mathcal{C}\mathbf{U}^\top}\left(\frac{c_i}{\|\mathbf{u}_i\|_2}\Big|_{\substack{i=1\\ i\neq t}}^{C}\right)$$

Proof. We observe that the Gaussian random variable $\frac{\mathbf{u}_i}{\|\mathbf{u}_i\|}{}^\top \epsilon|_{\substack{i=1\\ t\neq t}}^{C} = \mathbf{U}^\top\epsilon$ has mean zero as ϵ is mean zero. Computing its covariance matrix, we have $\mathbb{E}_\epsilon\, \mathbf{U}^\top\epsilon\epsilon^\top\mathbf{U} = \mathbf{U}^\top\, \mathbb{E}_\epsilon(\epsilon\epsilon^\top)\mathbf{U} = \mathbf{U}^\top\mathcal{C}\mathbf{U}$. We use this result after equation 2 in the proof of Lemma 1. □

Proposition A.1. *The **Taylor estimator** for the local robustness of a classifier f at point $\mathbf{x}$ with respect to target class t is given by linearizing f around $\mathbf{x}$ using a first-order Taylor expansion, with decision boundaries $g_i(\mathbf{x}) = f_t(\mathbf{x}) - f_i(\mathbf{x})$, $\forall i \neq t$, leading to*

$$p_\sigma^{taylor}(\mathbf{x}) = \Phi_{\mathbf{U}\mathbf{U}^\top}\left(\frac{g_i(\mathbf{x})}{\sigma\|\nabla_{\mathbf{x}} g_i(\mathbf{x})\|_2}\Big|_{\substack{i=1\\ i\neq t}}^{C}\right)$$

with $\mathbf{U}$ and Φ defined as in the linear case.

Proof. Using the notations from the previous Lemma A.1, we can linearize $g(\mathbf{x}+\epsilon) \approx g(\mathbf{x}) + \nabla_{\mathbf{x}} g(\mathbf{x})^\top\epsilon$ using a first order Taylor series expansion. Thus we use $\mathbf{u}_i = \nabla_{\mathbf{x}} g_i(\mathbf{x})$ and $c_i = g_i(\mathbf{x})$, and plug it into the result of Lemma A.1. □

Proposition A.2. *The **estimation error** of the Taylor estimator for a classifier with a quadratic decision boundary $g_i(\mathbf{x}) = \mathbf{x}^\top A_i\mathbf{x} + \mathbf{u}_i^\top\mathbf{x} + c_i$ for positive-definite A_i, is upper bounded by*

$$|p_\sigma^{robust}(\mathbf{x}) - p_\sigma^{taylor}(\mathbf{x})| \leq k\sigma^{C-1}\prod_{\substack{i=1\\ i\neq t}}^{C}\frac{\lambda_{\max}^{A_i}}{\|\mathbf{u}_i\|_2}$$

for noise $\epsilon \sim \mathcal{N}(0, \sigma^2/d)$, in the limit of $d \to \infty$.

Proof. Without loss of generality, assume that $\mathbf{x} = 0$. For any other $\mathbf{x}_1 \neq 0$, we can simply perform a change of variables of the underlying function to center it at $\mathbf{x}_1$ to yield a different quadratic. We first write an expression for p_σ^{robust} for the given quadratic classifier $g_i(\mathbf{x})$ at $\mathbf{x} = 0$.

$$\begin{aligned} p_\sigma^{robust}(0) &= P_\epsilon \left(\bigcup_i g_i(\epsilon) > 0 \right) \\ &= P_\epsilon \left(\bigcup_i \mathbf{u}_i^\top \epsilon + c > -\epsilon^\top A_i \epsilon \right) \end{aligned}$$

Similarly, computing, p_σ^{taylor} we have $\nabla_{\mathbf{x}} g_i(0) = \mathbf{u}^\top$ and $g_i(0) = c_i$, resulting in

$$\begin{aligned} p_\sigma^{taylor}(0) &= P_\epsilon \left(\bigcup_i g_i^{taylor}(\epsilon) > 0 \right) \\ &= P_\epsilon \left(\bigcup_i \mathbf{u}_i^\top \epsilon + c > 0 \right) \end{aligned}$$

Subtracting the two, we have

$$\begin{aligned} &|p_\sigma^{robust}(0) - p_\sigma^{taylor}(0)| \\ &= \left| P \left(\bigcup_i 0 > \mathbf{u}_i^\top \epsilon + c > -\epsilon^\top A_i \epsilon \right) \right| \\ &= \left| P \left(\bigcup_i 0 > \frac{\mathbf{u}_i^\top \epsilon + c}{\sigma \|\mathbf{u}_i\|_2} > -\frac{\epsilon^\top A_i \epsilon}{\sigma \|\mathbf{u}_i\|_2} \right) \right| \end{aligned}$$

For high-dimensional Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2/d)$, with $d \to \infty$, we have that $\|\epsilon\|^2 = \sum_i \epsilon_i^2 \to \sigma^2$ from the law of large numbers. See [Vershynin, 2018] for an extended discussion. Thus we have $\epsilon^\top A \epsilon \leq \lambda_{\max}^A \|\epsilon\|^2 = \lambda_{\max}^A \sigma^2$.

Also let $z_i = \frac{\mathbf{u}_i^\top \epsilon + c}{\sigma \|\mathbf{u}_i\|_2}$ be a random variable. We observe that $z_i|_i$ is a tensor extension of z_i, has a covariance matrix of $\mathbf{U}\mathbf{U}^\top$ as before. Let us also define $\mathcal{C}_i = \frac{\lambda_{\max}^{A_i}}{\|\mathbf{u}_i\|_2}$.

$$
\begin{aligned}
&|p_\sigma^{robust}(0) - p_\sigma^{taylor}(0)| \\
&= \left| P\left(\bigcup_i 0 > z_i(\epsilon) > -\frac{\epsilon^\top A_i \epsilon}{\sigma \|\mathbf{u}_i\|_2} \right) \right| \\
&\leq \left| P\left(\bigcup_i 0 > z_i > -\frac{\lambda_{\max}^{A_i}}{\|\mathbf{u}_i\|_2}\sigma \right) \right| \quad (\epsilon^\top A \epsilon < \lambda_{\max}^A \sigma^2) \\
&= \left| \int \ldots \int_{-\mathcal{C}_i\sigma}^{0} \text{pdf}(z_i|_i)\, \mathrm{d}z_i|_i \right| \quad \text{(Defn of mvn cdf)} \\
&\leq \max_{z_i|_i} \text{pdf}(z_i|_i) \prod_i |C_i \sigma| \quad \text{(Upper bound pdf with its max)} \\
&\leq (2\pi)^{-(C-1)/2} \det(\mathbf{U}\mathbf{U}^\top)^{-1/2} \prod_{\substack{i=1 \\ i\neq t}}^{C} C_i \sigma \\
&= k \left(\sigma^{C-1} \prod_{\substack{i=1 \\ i\neq t}}^{C} \frac{\lambda_{\max}^{A_i}}{\|\mathbf{u}_i\|_2} \right)
\end{aligned}
$$

where $k = \max_z pdf(z) = (2\pi)^{-(C-1)/2} \det(\mathbf{U}\mathbf{U}^\top)^{-1/2}$, which is the max value of the Gaussian pdf. Note that as the rows of $\mathbf{U}$ are normalized, $\det(\mathbf{U}) \leq 1$ and $\det(\mathbf{U}\mathbf{U}^\top) = \det(\mathbf{U})^2 \leq 1$.

□

We note that these bounds are rather pessimistic, as in high-dimensions $\epsilon^\top A_i \epsilon \sim \lambda_{\text{mean}}^{A_i} \leq \lambda_{\max}^{A_i}$, and thus in reality the errors are expected to be much smaller.

Proposition A.3. *The **MMSE estimator** for the local robustness of a classifier f at point $\mathbf{x}$ with respect to target class t is given by an MMSE linearization f around $\mathbf{x}$, for decision boundaries $g_i(\mathbf{x}) = f_t(\mathbf{x}) - f_i(\mathbf{x})$, $\forall i \neq t$, leading to*

$$
\begin{aligned}
p_\sigma^{mmse}(\mathbf{x}) &= \Phi_{\mathbf{U}\mathbf{U}^\top} \left(\frac{\tilde{g}_i(\mathbf{x})}{\sigma \|\nabla_{\mathbf{x}} \tilde{g}_i(\mathbf{x})\|_2} \Big|_{\substack{i=1 \\ i\neq t}}^{C} \right) \\
\text{where } \tilde{g}_i(\mathbf{x}) &= \frac{1}{N} \sum_{j=1}^{N} g_i(\mathbf{x} + \epsilon)\ ,\ \epsilon \sim \mathcal{N}(0, \sigma^2)
\end{aligned}
$$

with $\mathbf{U}$ and Φ defined as in the linear case, and N is the number of perturbations.

Proof. We would like to improve upon the Taylor approximation to $g(\mathbf{x}+\epsilon)$ by using an MMSE local function approximation. Essentially, we'd like the find $\mathbf{u} \in \mathbb{R}^d$ and $c \in \mathbb{R}$ such that

$$
(\mathbf{u}^*(\mathbf{x}), c^*(\mathbf{x})) = \arg\min_{\mathbf{u},c} \mathop{\mathbb{E}}_{\epsilon \sim \mathcal{N}(0,\sigma^2)} (g(x+\epsilon) - \mathbf{u}^\top \epsilon - c)^2
$$

A straightforward solution by finding critical points and equating it to zero gives us the following:

$$
\begin{aligned}
\mathbf{u}^*(\mathbf{x}) &= \mathop{\mathbb{E}}_{\epsilon}\left[g(x+\epsilon)\epsilon^\top\right]/\sigma^2 \\
&= \mathop{\mathbb{E}}_{\epsilon}\left[\nabla_{\mathbf{x}} g(\mathbf{x}+\epsilon)\right] \qquad \text{(Stein's Lemma)} \\
c^*(\mathbf{x}) &= \mathop{\mathbb{E}}_{\epsilon} g(x+\epsilon)
\end{aligned}
$$

Plugging in these values of U^*, c^* into Lemma A.1, we have the result.

□

Proposition A.4. *The **estimation error** of the MMSE estimator for a classifier with a quadratic decision boundary $g(\mathbf{x}) = \mathbf{x}^\top A\mathbf{x} + \mathbf{u}^\top\mathbf{x} + c$, and positive definite A is upper bounded by*

$$
|p_\sigma^{robust}(\mathbf{x}) - p_\sigma^{mmse}(\mathbf{x})| \leq k\sigma^{C-1}\prod_{\substack{i=1\\ i\neq t}}^{C}\frac{|\lambda_{\max}^{A_i} - \lambda_{\text{mean}}^{A_i}|}{\|\mathbf{u}_i\|_2}
$$

for noise $\epsilon \sim \mathcal{N}(0, \sigma^2/d)$, in the limit of $d \to \infty$ and $N \to \infty$.

Proof. We proceed similarly to the proof made for the Taylor estimator, and without loss of generality, assume that $\mathbf{x} = 0$. Computing, p_σ^{mmse} we have $\mathbb{E}_\epsilon \nabla_{\mathbf{x}} g_i(\epsilon) = \mathbf{u}_i^\top$ and $\mathbb{E}_\epsilon\, g_i(\epsilon) = c + \mathbb{E}(\epsilon^\top A_i \epsilon) = c + \mathbb{E}(trace(\epsilon^\top A_i \epsilon)) = c + \mathbb{E}(trace(A_i \epsilon\epsilon^T)) = c + trace(A_i)\sigma^2/d = c + \sigma^2\lambda_{\text{mean}}^{A_i}$, resulting in

$$
\begin{aligned}
p_\sigma^{mmse}(0) &= P_\epsilon\left(\bigcup_i \hat{g}_i(\epsilon) > 0\right) \\
&= P_\epsilon\left(\bigcup_i \mathbf{u}_i^\top\epsilon + c > -\sigma^2\lambda_{\text{mean}}^{A_i}\right)
\end{aligned}
$$

Subtracting the two, we have

$$
\begin{aligned}
&|p_\sigma^{robust}(0) - p_\sigma^{mmse}(0)| \\
&\leq \left|P\left(\bigcup_i -\sigma^2\lambda_{\text{mean}}^{A_i} > \mathbf{u}_i^\top\epsilon + c > -\sigma^2\lambda_{\max}^{A_i}\right)\right| \\
&= \left|P\left(\bigcup_i -\sigma\frac{\lambda_{\text{mean}}^{A_i}}{\|\mathbf{u}_i\|_2} > \frac{\mathbf{u}_i^\top\epsilon + c}{\sigma\|\mathbf{u}_i\|_2} > -\sigma\frac{\lambda_{\max}^{A_i}}{\|\mathbf{u}_i\|_2}\right)\right|
\end{aligned}
$$

Similar to the previous proof, let $z_i = \mathbf{u}_i^\top\epsilon + c$ be a random variable, and that $z_i|_i$ is a tensor extension of z_i from our previous notation.

$$
\begin{aligned}
&|p_\sigma^{robust}(\mathbf{x}) - p_\sigma^{mmse}(\mathbf{x})| \\
&\leq \left| P\left(\bigcup_i -\lambda_{\text{mean}}^{A_i}\sigma^2 > z_i > -\lambda_{\max}^{A_i}\sigma^2 \right) \right| \\
&= \left| \int ... \int_{-\lambda_{\max}^{A_i}\sigma^2}^{-\lambda_{\text{mean}}^{A_i}\sigma^2} \text{pdf}(z_i|_i)\, \mathrm{d}z_i|_i \right| \\
&\leq \max_{z_i|_i} \text{pdf}(z_i|_i)\, \sigma^{C-1} \prod_i \frac{|(\lambda_{\max}^{A_i} - \lambda_{\text{mean}}^{A_i})|}{\|\mathbf{u}_i\|_2} \\
&= k\sigma^{C-1} \prod_i \frac{|\lambda_{\max}^{A_i} - \lambda_{\text{mean}}^{A_i}|}{\|\mathbf{u}_i\|2}
\end{aligned}
$$

where $k = \max_z \text{pdf}(z_i|_i) = (2\pi)^{-(C-1)/2} \det(\mathbf{U}\mathbf{U}^\top)^{-1/2}$ like in the Taylor case. Note that as the rows of $\mathbf{U}$ are normalized, $\det(\mathbf{U}) \leq 1$ and $\det(\mathbf{U}\mathbf{U}^\top) = \det(\mathbf{U})^2 \leq 1$.

□

We note that these bounds are rather pessimistic, as in high-dimensions $\epsilon^\top A_i \epsilon \sim \lambda_{\text{mean}}^{A_i} \leq \lambda_{\max}^{A_i}$, and thus in reality the errors are expected to be much smaller.

A.1.1 Approximating the Multivariate Gaussian CDF with mv-sigmoid

One drawback of the Taylor and MMSE estimators is their use of the *mvn-cdf*, which does not have a closed form solution and can cause the estimators to be slow for settings with a large number of classes C. In addition, the *mvn-cdf* makes these estimators non-differentiable, which is inconvenient for applications which require differentiating p_σ^{robust}. To alleviate these issues, we approximate the *mvn-cdf* with an analytical closed-form expression. As CDFs are monotonically increasing functions, the approximation should also be monotonically increasing.

To this end, it has been previously shown that the *univariate* Normal CDF ϕ is well-approximated by the sigmoid function [Hendrycks and Gimpel, 2016]. It is also known that when $\mathbf{U}\mathbf{U}^\top = I$, *mvn-cdf* is given by $\Phi(\mathbf{x}) = \prod_i \phi(\mathbf{x}_i)$, i.e., it is given by the product of the univariate normal CDFs. Thus, we may choose to approximate $\Phi(\mathbf{x}) = \prod_i \text{sigmoid}(\mathbf{x})$. However, when the inputs are small, this can be simplified as follows:

$$
\begin{aligned}
\Phi_I(\mathbf{x}) &= \prod_i \phi(\mathbf{x}_i) \approx \prod_i \frac{1}{1+\exp(-\mathbf{x}_i)} \\
&= \frac{1}{1 + \sum_i \exp(-\mathbf{x}_i) + \sum_{j,k} \exp(-\mathbf{x}_j - \mathbf{x}_k) + ...} \\
&\approx \frac{1}{1 + \sum_i \exp(-\mathbf{x}_i)} \quad (\text{for } \mathbf{x}_i \to \infty \ \forall i)
\end{aligned}
$$

We call the final expression the "multivariate sigmoid" (*mv-sigmoid*) which serves as our approximation of *mvn-cdf*, especially at the tails of the distribution. While we expect estimators using *mv-sigmoid* to approximate ones using *mvn-cdf* only when $\mathbf{U}\mathbf{U}^\top = \mathbf{I}$, we find experimentally that the approximation works well even for practical values of the covariance matrix $\mathbf{U}\mathbf{U}^\top$. Using this approximation to substitute *mv-sigmoid* for *mvn-cdf* in the p_σ^{taylor} and p_σ^{mmse} estimators yields the $p_\sigma^{\text{taylor\_mvs}}$ and $p_\sigma^{\text{mmse\_mvs}}$ estimators, respectively.

A.1.2 Relationship between mv-sigmoid, softmax, and the Taylor estimator

A common method to estimate the confidence of model predictions is to use the softmax function applied to the logits $f_i(\mathbf{x})$ of a model. We note that softmax is identical to *mv-sigmoid* when directly applied to the logits of neural networks:

$$\text{softmax}_t\left(f_i(\mathbf{x})\,\Big|_{i=1}^{C}\right) = \frac{\exp(f_t(\mathbf{x}))}{\sum_{i=1}^{C}\exp(f_i(\mathbf{x}))} = \frac{1}{1+\sum\limits_{\substack{i=1\\ i\neq t}}^{C}\exp(f_i(\mathbf{x})-f_t(\mathbf{x}))} = \text{mv-sigmoid}\left(g_i(\mathbf{x})\,\Big|_{\substack{i=1\\ i\neq t}}^{C}\right)$$

Recall that $g_i(\mathbf{x}) = f_t(\mathbf{x}) - f_i(\mathbf{x})$ is the decision boundary function. Note that this equivalence only holds for the specific case of logits. Comparing the expressions of softmax applied to logits above and the Taylor estimator, we notice that they are only different in that the Taylor estimator divides by the gradient norm, and uses the *mvn-cdf* function instead of *mv-sigmoid*. Given this similarity to the Taylor estimator, it is reasonable to ask whether softmax applied to logits (henceforth p_T^{softmax} for softmax with temperature T) itself can be a "good enough" estimator of p_σ^{robust} in practice. In other words, does p_T^{softmax} well-approximate p_σ^{robust} in certain settings?

In general, this cannot hold because softmax does not take in information about $\mathbf{U}\mathbf{U}^\top$, nor does it use the gradient information used in all of our estimators, although the temperature parameter T can serve as a substitute for σ in our expressions. In Appendix A.1, we provide a theoretical result for a restricted linear setting where softmax can indeed match the behavior of $p_\sigma^{\text{taylor\_mvs}}$, which happens precisely when $\mathbf{U}\mathbf{U}^\top = \mathbf{I}$ and all the class-wise gradients are equal. In the next section, we demonstrate empirically that the softmax estimator p_T^{softmax} is a poor estimator of average robustness in practice.

The softmax estimator We observe that for linear models with a specific noise perturbation σ, the common softmax function taken with respect to the output logits can be viewed as an estimator of p_σ^{robust}, albeit in a very restricted setting. Specifically,

Lemma A.4. *For multi-class linear models $f(\mathbf{x}) = \mathbf{w}^\top\mathbf{x} + b$, such that the decision boundary weight norms $\|\mathbf{u}_i\|_2 = k, \forall i \in [1, C], i \neq t$,*

$$p_T^{\textit{softmax}} = p_\sigma^{\textit{taylor\_mvs}} \quad \textit{where} \quad T = \sigma k$$

Proof. Consider softmax with respect to the t^{th} output class and define $g_i(\mathbf{x}) = f_t(\mathbf{x}) - f_i(\mathbf{x})$, with f being the linear model logits. Using this, we first show that softmax is identical to *mv-sigmoid*:

$$\begin{aligned} p_T^{\text{softmax}}(\mathbf{x}) &= \text{softmax}_t(f_1(\mathbf{x})/T, ..., f_C(\mathbf{x})/T) \\ &= \frac{\exp(f_t(\mathbf{x})/T)}{\sum_i \exp(f_i(\mathbf{x})/T)} \\ &= \frac{1}{1+\sum_{i;i\neq t}\exp((f_i(\mathbf{x})-f_t(\mathbf{x}))/T)} \\ &= \text{mv-sigmoid}\left[g_i(\mathbf{x})/T\,\Big|_{\substack{i=1\\ i\neq t}}^{C}\right] \end{aligned}$$

Next, by denoting $\mathbf{u}_i = \mathbf{w}_t - \mathbf{w}_i$, each row has equal norm $\|\mathbf{u}_i\|_2 = \|\mathbf{u}_j\|_2, \forall i, j, t \in [1, ...C]$ which implies:

$$\begin{aligned} p_\sigma^{\text{taylor\_mvs}}(\mathbf{x}) &= \text{mv-sigmoid}\left[\frac{g_i(\mathbf{x})}{\sigma\|\mathbf{u}_i\|_2}\,\Big|_{\substack{i=1\\ i\neq t}}^{C}\right] \\ &= \text{mv-sigmoid}\left[g_i(\mathbf{x})/T\,\Big|_{\substack{i=1\\ i\neq t}}^{C}\right] \quad (\because T = \sigma k) \\ &= p_T^{\text{softmax}}(\mathbf{x}) \end{aligned}$$

□

Lemma A.4 indicates that the temperature parameter T of softmax roughly corresponds to the σ of the added Normal noise with respect to which local robustness is measured. Overall, this shows that under the restricted setting where the local linear model consists of decision boundaries with equal weight norms, the softmax outputs can be viewed as an estimator of the $p_\sigma^{\text{taylor\_mvs}}$ estimator, which itself is an estimator of p_σ^{robust}. However, due to the multiple levels of approximation, we can expect the quality of p_T^{softmax}'s approximation of p_σ^{robust} to be poor in general settings (outside of the very restricted setting), so much so that in general settings, p_σ^{robust} and p_T^{softmax} would be unrelated.

A.2 DATASETS

The MNIST dataset consists of images of gray-scale handwritten digits spanning 10 classes: digits 0 through 9. The FashionMNIST (FMNIST) dataset consists of gray-scale images of articles of clothing spanning 10 classes: t-shirt, trousers, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot. For MNIST and FMNIST, each image is 28 pixels x 28 pixels. For MNIST and FMNIST, the training set consists of 60,000 images and the test set consists of 10,000 images.

The CIFAR10 dataset consists of color images of common objects and animals spanning 10 classes: airplane, car, bird, cat, deer, dog, frog, horse, ship, and truck. The CIFAR100 dataset consists of color images of common objects and animals spanning 100 classes: apple, bowl, chair, dolphin, lamp, mouse, plain, rose, squirrel, train, etc. For CIFAR10 and CIFAR100, each image is 3 pixels x 32 pixels x 32 pixels. For CIFAR10 and CIFAR100, the training set consists of 50,000 images and the test set consists of 10,000 images.

A.3 MODELS

For the MNIST and FMNIST, we train a linear model and a convolutional neural network (CNN) to perform 10-class classification. The linear model consists of one hidden layer with 10 neurons. The CNN consists of four hidden layers: one convolutional layer with 5x5 filters and 10 output channels, one convolutional layer 5x5 filters and 20 output channels, and one linear layer with 50 neurons, and one linear layer 10 neurons.

For CIFAR10 and CIFAR100, we train a Vision Transformer model to perform 10-class and 100-class classification, respectively, by fine-tuning a Vision Transformer that was pre-trained on ImageNet (`https://huggingface.co/google/vit-base-patch16-224-in21k`) on each dataset. For these models, the test set consists of 100 images. We chose this number of datapoints so that p_σ^{mc} would run within a reasonable amount of time. We also train a ResNet18 model to perform 10-class and 100-class classification, respectively. The model architecture is described in [He et al., 2016]. For CIFAR10 and CIFAR100, we also train the ResNet18 models using varying levels of gradient norm regularization to obtain models with varying levels of robustness. The larger the weight of gradient norm regularization (λ), the more robust the model.

All models were trained using stochastic gradient descent. Hyperparameters were selected to achieve decent model performance. The emphasis is on analyzing the estimators' estimates of local robustness of each model, not on high model performance. Thus, we do not focus on tuning model hyperparameters. All models were trained for 200 epochs. The test set accuracy for each model is shown in Table 2.

EXPERIMENTS

Due to file size constraints, Section A.4 can be found in the Supplementary material.

Dataset	Model	λ	Test set accuracy
MNIST	Linear	0	92%
MNIST	CNN	0	99%
FashionMNIST	Linear	0	84%
FashionMNIST	CNN	0	91%
CIFAR10	Vision Transformer	0	99%
CIFAR10	ResNet18	0	94%
CIFAR10	ResNet18	0.0001	93%
CIFAR10	ResNet18	0.001	90%
CIFAR10	ResNet18	0.01	85%
CIFAR100	Vision Transformer	0	91%
CIFAR100	ResNet18	0	76%
CIFAR100	ResNet18	0.0001	74%
CIFAR100	ResNet18	0.001	69%
CIFAR100	ResNet18	0.01	60%

Table 2: Test set accuracy of models.

A.4 EXPERIMENTS

In this section, we provide the following additional experimental results:

1. Figure 5 shows results on the convergence of p_σ^{mc}. p_σ^{mc} takes a large number of samples to converge and is computationally inefficient.
2. Figure 6 shows results on the convergence of p_σ^{mmse}. p_σ^{mmse} takes only a few samples to converge and is more computationally inefficient than p_σ^{mc}.
3. Figure 7 shows the distribution of p_σ^{robust} as a function of σ. Consistent with theory in Section 3, (1) as noise increases, p_σ^{robust} decreases, and (2) p_σ^{mmse} accurately estimates p_σ^{mc}.
4. Table 3 presents estimator runtimes. Our analytical estimators are more efficient than the naïve estimator (p_σ^{mc}).
5. Figure 8 shows the accuracy of the analytical robustness estimators as a function of σ. p_σ^{mmse} and $p_\sigma^{\text{mmse\_mvs}}$ are the best estimators of p_σ^{robust}, followed closely by $p_\sigma^{\text{taylor\_mvs}}$ and p_σ^{taylor}, trailed by p_T^{softmax}.
6. Figure 9 shows the accuracy of the analytical estimators for robust models. For more robust models, the estimators compute p_σ^{robust} more accurately over a larger σ.
7. Figures 10 and 11 shows that *mv-sigmoid* well-approximates *mvn-cdf* over σ.
8. Figure 12 shows that p_T^{softmax} is not a good approximator of p_σ^{robust}.
9. Figure 13 shows the distribution of p_σ^{robust} among classes (measured by p_σ^{mmse}), revealing that models display robustness bias among classes.
10. Figures 14 and 15 show the application of p_σ^{mmse} and p_T^{softmax} to identification of robust and non-robust points. p_σ^{robust} better identifies robust and non-robust points than p_T^{softmax}.
11. Figures 16, 17, 18, and 19 show examples of noisy images with the level of noise analyzed in our paper. Overall, the noise levels seem visually significant.

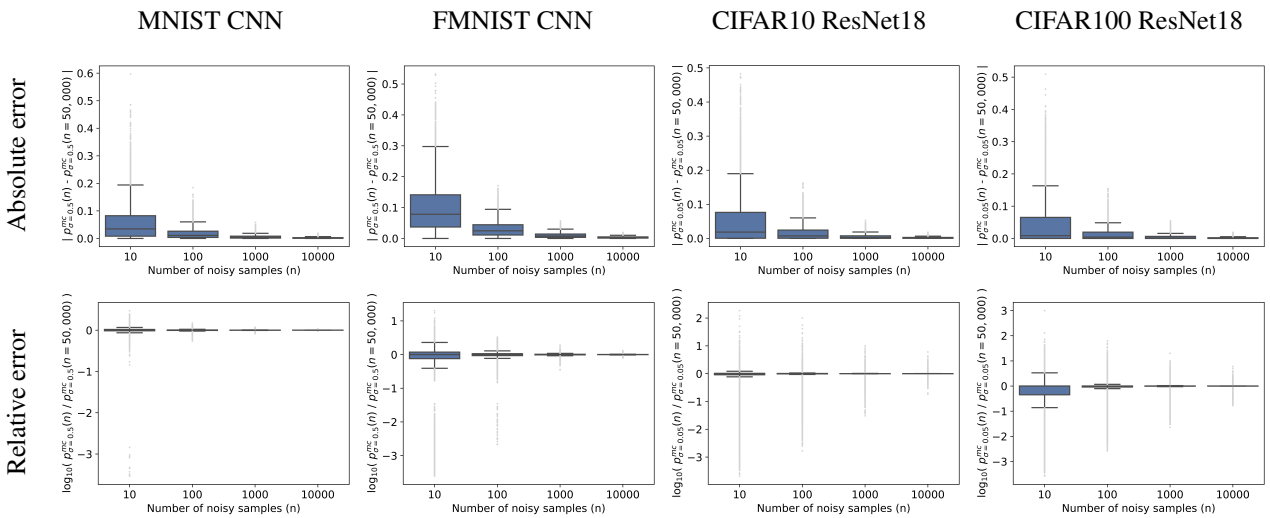


Figure 5: Convergence of p_σ^{mc}. In practice, p_σ^{mc} takes around $n = 10,000$ samples to converge and is computationally inefficient.

A.4.1 p_σ^{robust} identifies images that are robust to and images that are vulnerable to random noise

For each dataset, we train a simple CNN to distinguish between images with high and low p_σ^{mmse}. We train the same CNN to also distinguish between images with high and low p_T^{softmax}. The CNN consists of two convolutional layers and two fully-connected feedforward layers with a total of 21,878 parameters. For a given dataset, for each class, we take the images with the top-25 and bottom-25 p_σ^{mmse} values. This yields 500 images for CIFAR10 (10 classes x 50 images per class) and 5,000 images for CIFAR100 (100 classes x 50 images per class). We also perform the same steps using p_T^{softmax}, yielding

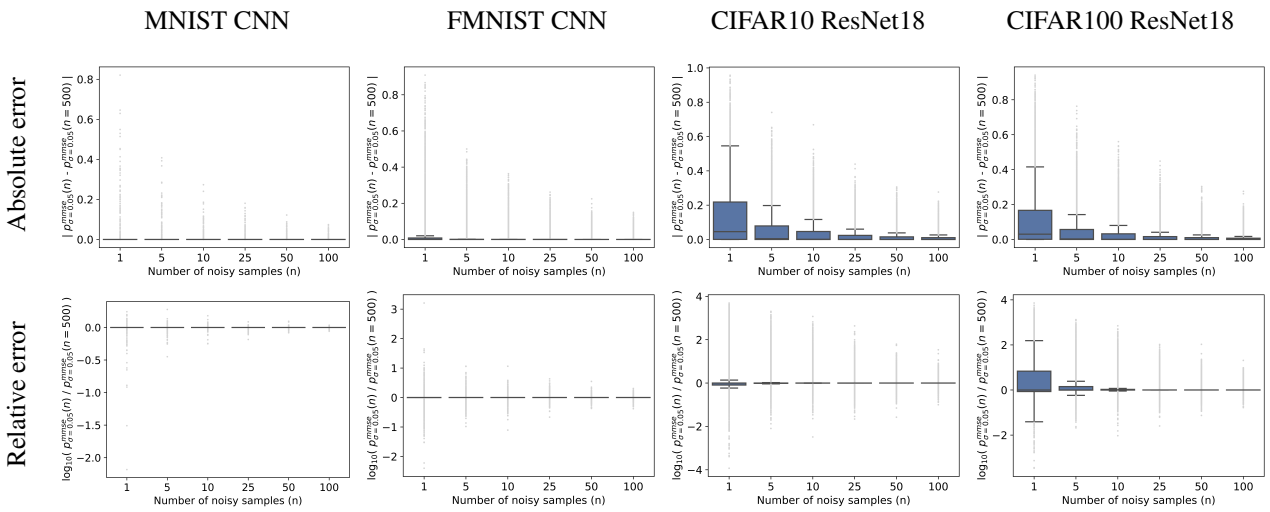
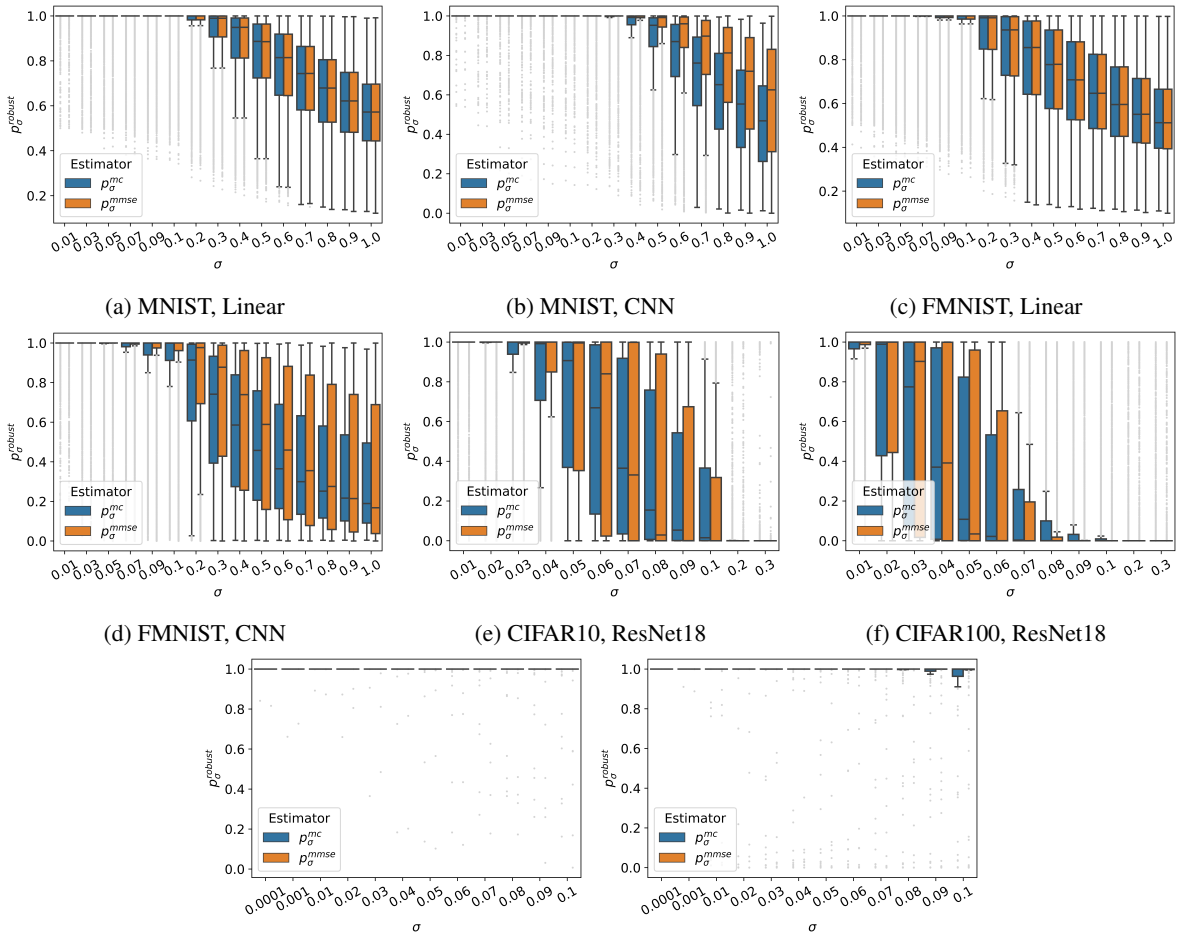


Figure 6: Convergence of p_σ^{mmse}. In practice, p_σ^{mmse} takes around $n = 5{-}10$ samples to converge and is more computationally efficient than p_σ^{mc}.

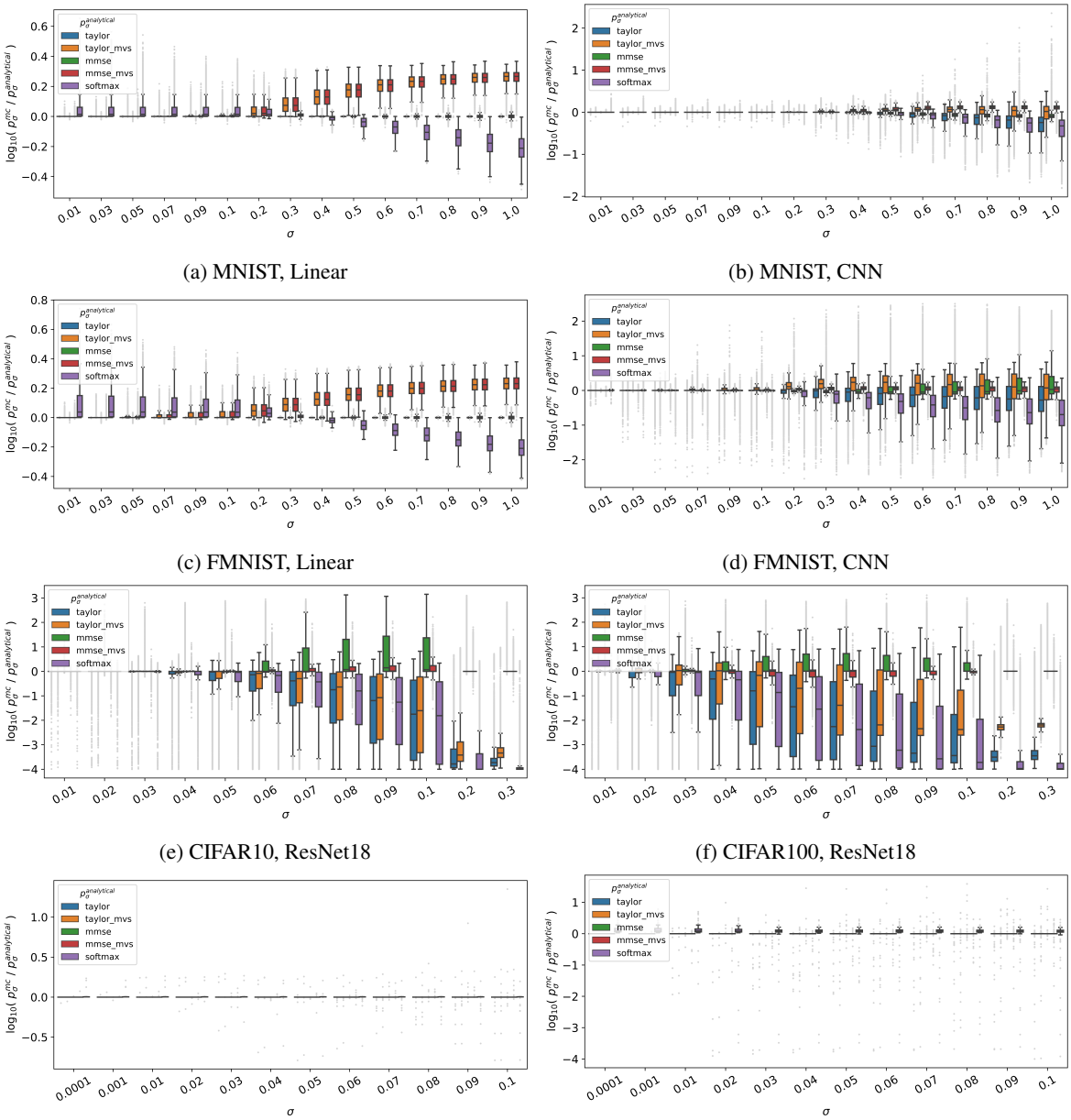


(a) MNIST, Linear (b) MNIST, CNN (c) FMNIST, Linear

(d) FMNIST, CNN (e) CIFAR10, ResNet18 (f) CIFAR100, ResNet18

(g) CIFAR10, Vision Transformer (h) CIFAR100, Vision Transformer

Figure 7: Distribution of p_σ^{robust} over σ. As noise increases, p_σ^{robust} decreases. In addition, p_σ^{mmse} accurately estimates p_σ^{mc}.



(a) MNIST, Linear

(b) MNIST, CNN

(c) FMNIST, Linear

(d) FMNIST, CNN

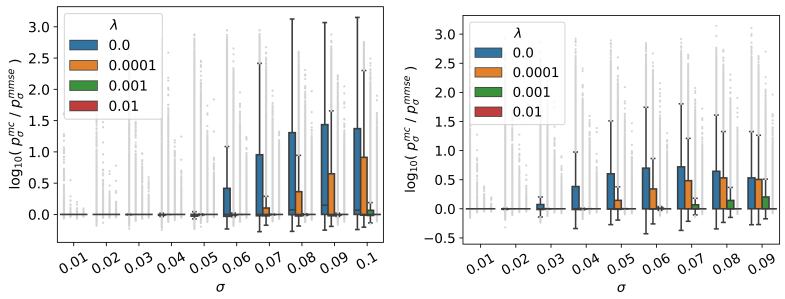
(e) CIFAR10, ResNet18

(f) CIFAR100, ResNet18

(g) CIFAR10, Vision Transformer

(h) CIFAR100, Vision Transformer

Figure 8: Accuracy of p_σ^{robust} estimators over σ. The smaller the noise neighborhood σ, the more accurately the estimators compute p_σ^{robust}. p_σ^{mmse} and $p_\sigma^{\text{mmse\_mvs}}$ are the best estimators of p_σ^{robust}, followed closely by $p_\sigma^{\text{taylor\_mvs}}$ and p_σ^{taylor}, trailed by p_T^{softmax}.



(a) CIFAR10, ResNet18 (b) CIFAR100, ResNet18

Figure 9: Accuracy of p_σ^{robust} estimators over σ for robust models. For more robust models, the estimators compute p_σ^{robust} more accurately over a larger σ.

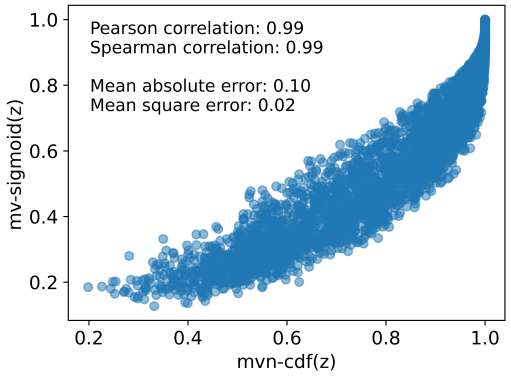


Figure 10: Correlation of *mvn-cdf(z)* and *mv-sigmoid(z)* for the CIFAR10 ResNet18 model. The formulation of z is described in Section 4.1. In practice, *mv-sigmoid* approximates *mvn-cdf* well.

another 500 images for CIFAR10 and another 5,000 images for CIFAR100. For each dataset, the train/test split is 90%/10% of points.

Then, we compare the performance of the two models. For CIFAR10, the test set accuracy for the p_σ^{mmse} CNN is 0.92 while that for the p_T^{softmax} CNN is 0.58. For CIFAR100, the test set accuracy for the p_σ^{mmse} CNN is 0.74 while that for the p_T^{softmax} CNN is 0.55. The higher the test set accuracy of a CNN, the better the CNN distinguishes between images. Thus, the results indicate that p_σ^{robust} better identifies images that are robust to and vulnerable to random noise than p_T^{softmax}.

We also provide additional visualizations of images with the highest and lowest p_σ^{robust} and images with the highest and lowest p_T^{softmax}.

A.4.2 Softmax probability is not a good proxy for average-case robustness

To examine the relationship between p_σ^{robust} and p_T^{softmax}, we calculate p_σ^{mmse} and p_T^{softmax} for CIFAR10 and CIFAR100 models of varying levels of robustness, and measure the correlation of their values and ranks using Pearson and Spearman correlations. Results are in Appendix A.4 (Figure 12). For a non-robust model, p_σ^{robust} and p_T^{softmax} are not strongly correlated (Figure 12a). As model robustness increases, the two quantities become more correlated (Figures 12b and 12c). However, even for robust models, the relationship between the two quantities is mild (Figure 12c). That p_σ^{robust} and p_T^{softmax} are not strongly correlated is consistent with the theory in Section 3: in general settings, p_T^{softmax} is not a good estimator for p_σ^{robust}.

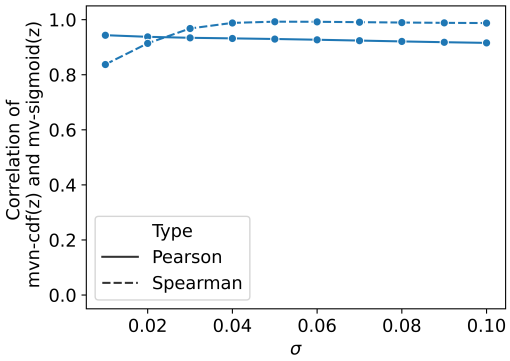
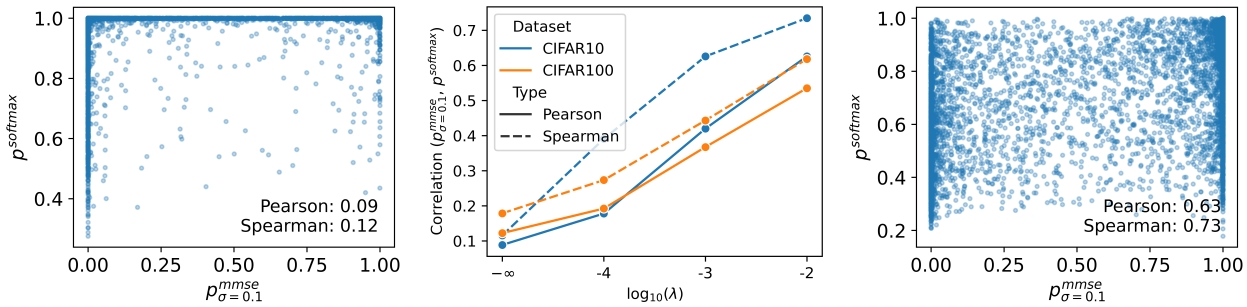


Figure 11: mv-sigmoid's approximation of mvn-cdf over σ. mv-sigmoid well-approximates mvn-cdf over σ.



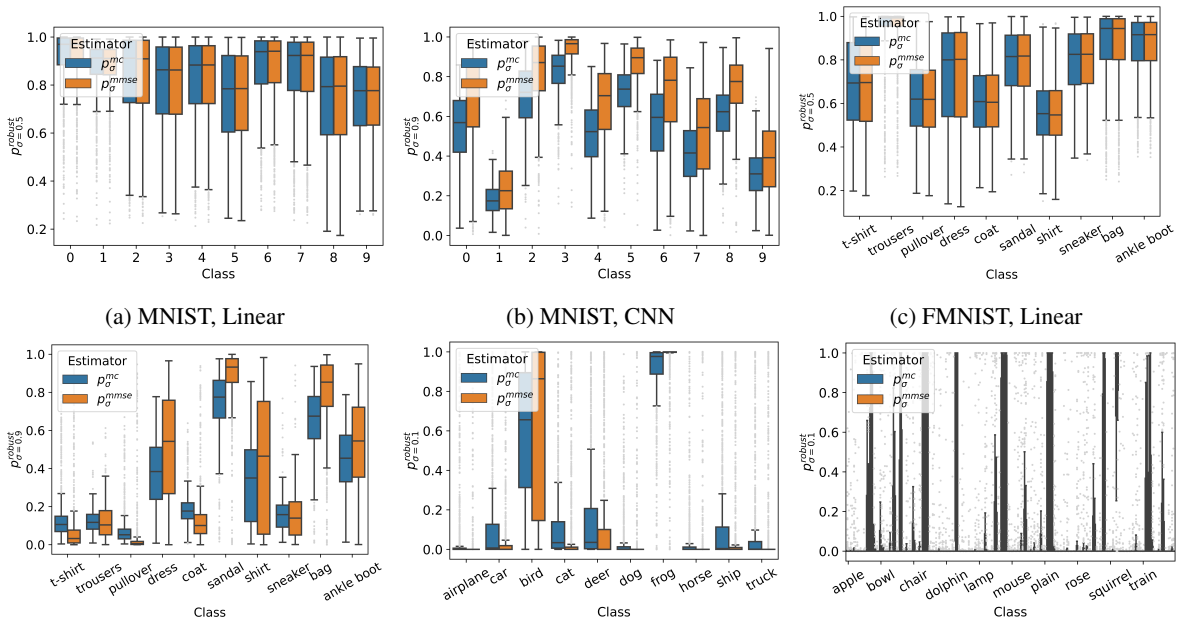
(a) CIFAR10
Non-robust model ($\lambda = 0$)

(b) CIFAR10 and CIFAR100
Varying model robustness

(c) CIFAR10
Robust model ($\lambda = 0.01$)

Figure 12: Relationship between p_σ^{robust} and p_T^{softmax} for CIFAR10 and CIFAR100 ResNet18 models. (a) For a non-robust model, p_σ^{robust} and p_T^{softmax} are not strongly correlated. (b) As model robustness increases, the two quantities become more correlated. (c) However, even for robust models, the relationship between p_σ^{robust} and p_T^{softmax} is mild. Together, these results indicate that, consistent with the theory in Section 3, p_T^{softmax} is not a good estimator for p_σ^{robust} in general settings.

While working on the paper, we hypothesized that p_σ^{robust} (e.g., p_σ^{mmse}) might be correlated with model accuracy. However, we did not find this in practice. Instead, what we find is that p_σ^{robust} succeeds in identifying canonical data points of a class, and does so much better than p_T^{softmax}. We first assess this finding through visual inspection, finding that images with higher p_σ^{robust} tend to be more canonical and clear images, and that this distinction is less apparent for p_T^{softmax} (Figures 3 and 14). We then use a model to classify these images as an additional, more objective assessment of this pattern (as discussed in Section 4.3).



(a) MNIST, Linear (b) MNIST, CNN (c) FMNIST, Linear

(d) FMNIST, CNN (e) CIFAR10, ResNet18 (f) CIFAR100, ResNet18

Figure 13: Local robustness bias among classes. p_σ^{robust} reveals that the model is less locally robust for some classes than for others. The analytical estimator p_σ^{mmse} properly captures this model bias.

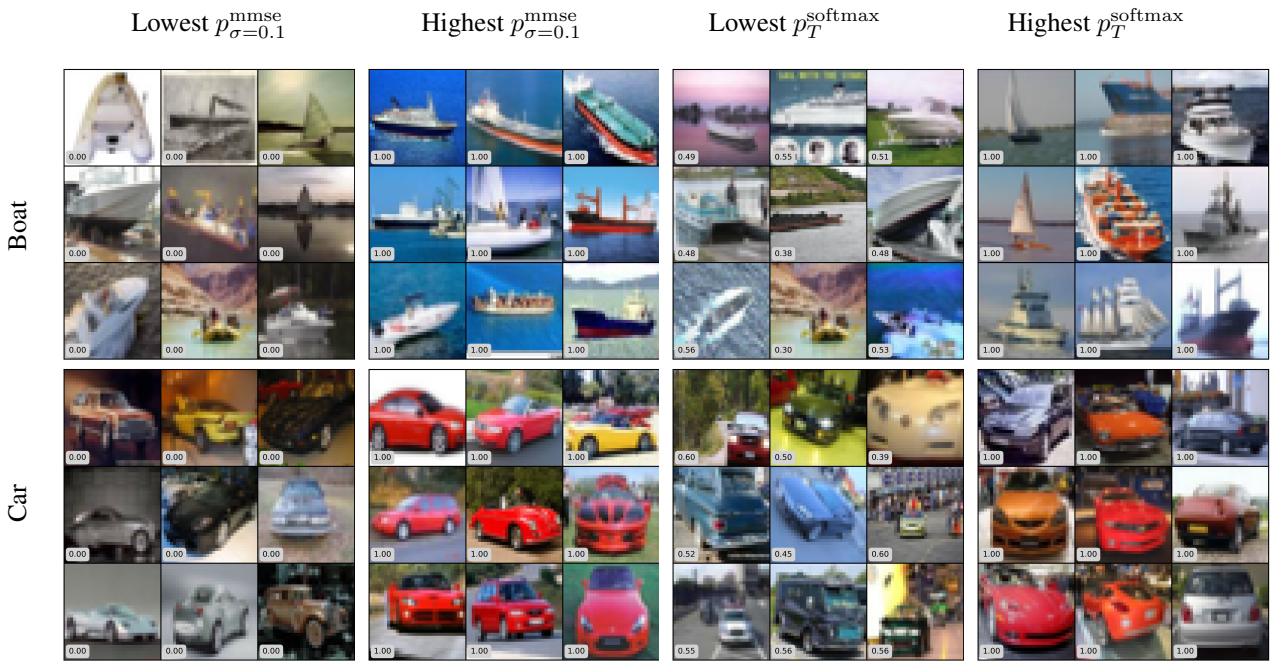


Figure 14: Additional images with the lowest and highest p_σ^{robust} and p_T^{softmax} values among CIFAR10 classes. Images with high p_σ^{robust} tend to be brighter and have stronger object-background contrast (making them more robust to random noise) than those with low p_σ^{robust}. The difference between images with high and low p_T^{softmax} is less clear. Thus, p_σ^{robust} better captures the model's local robustness with respect to an input than p_T^{softmax}.

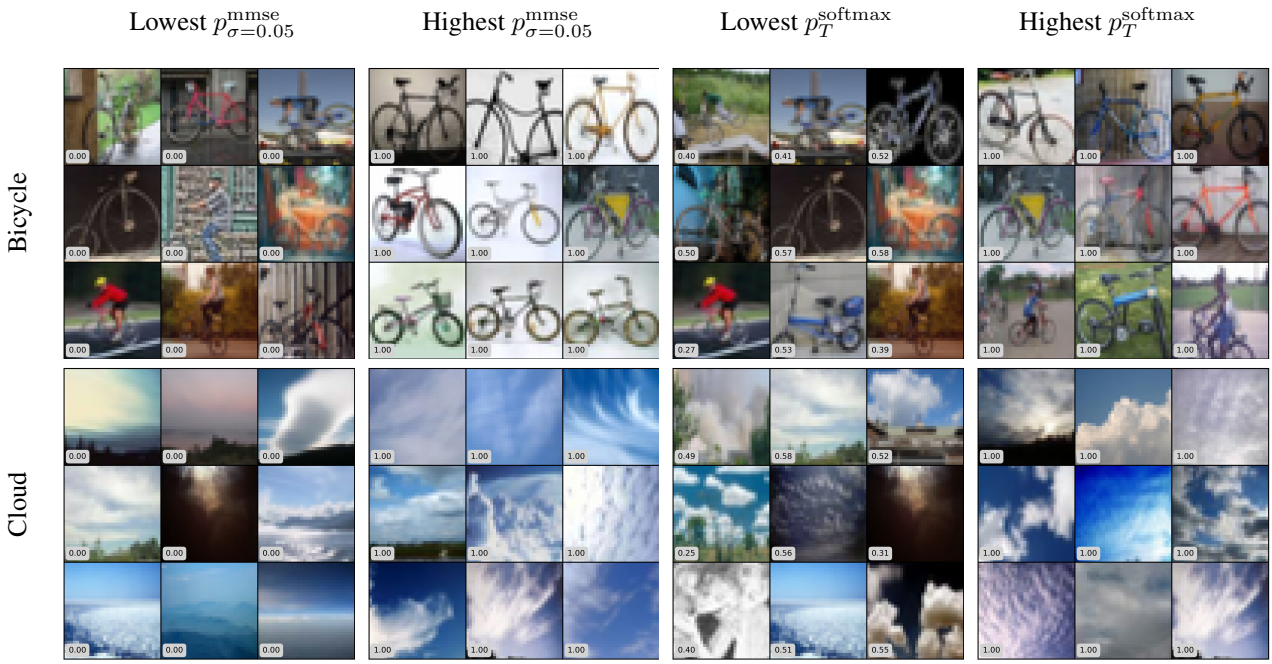


Figure 15: Images with the lowest and highest p_σ^{robust} and p_T^{softmax} values among CIFAR100 classes. Images with high p_σ^{robust} tend to be brighter and have stronger object-background contrast (making them more robust to random noise) than those with low p_σ^{robust}. The difference between images with high and low p_T^{softmax} is less clear. Thus, p_σ^{robust} better captures the model's local robustness with respect to an input than p_T^{softmax}.

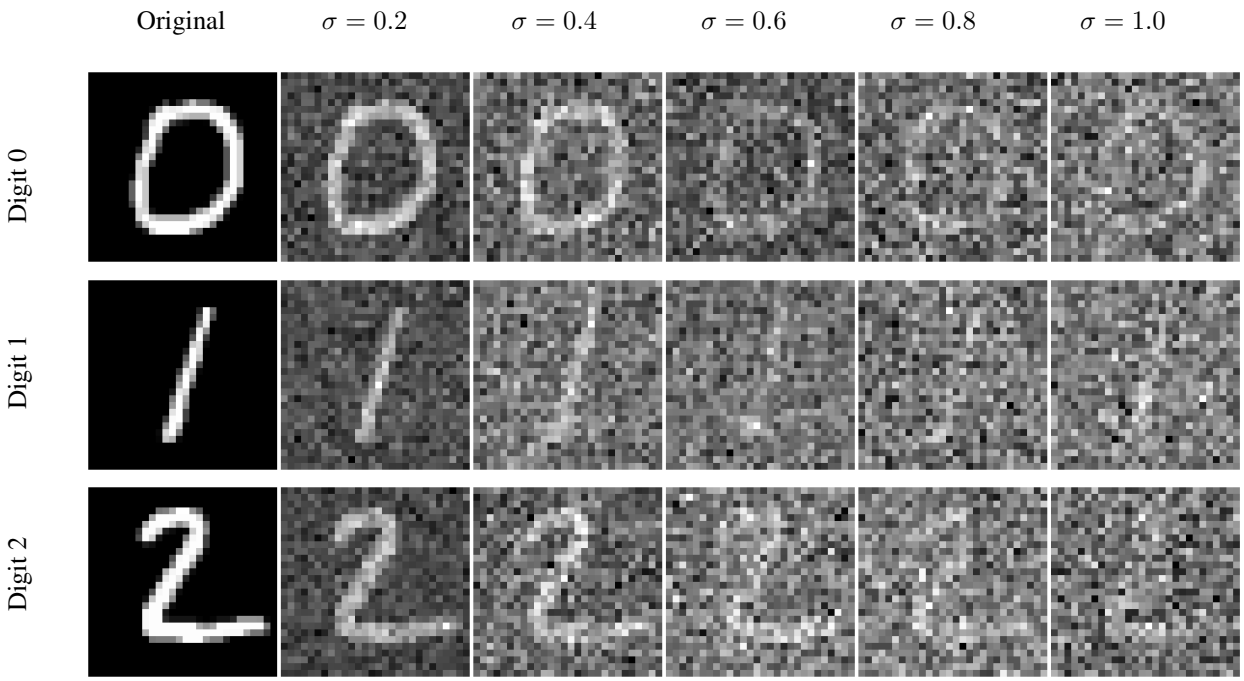


Figure 16: Examples of noisy images for MNIST.

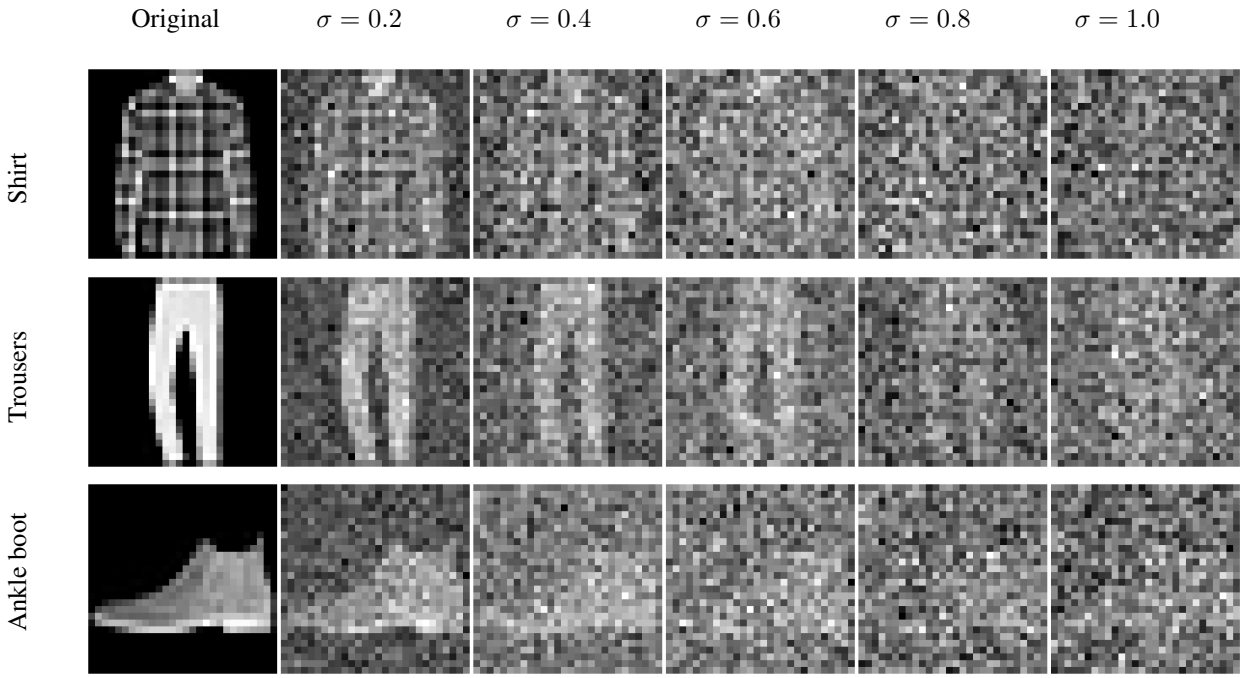


Figure 17: Examples of noisy images for FMNIST.

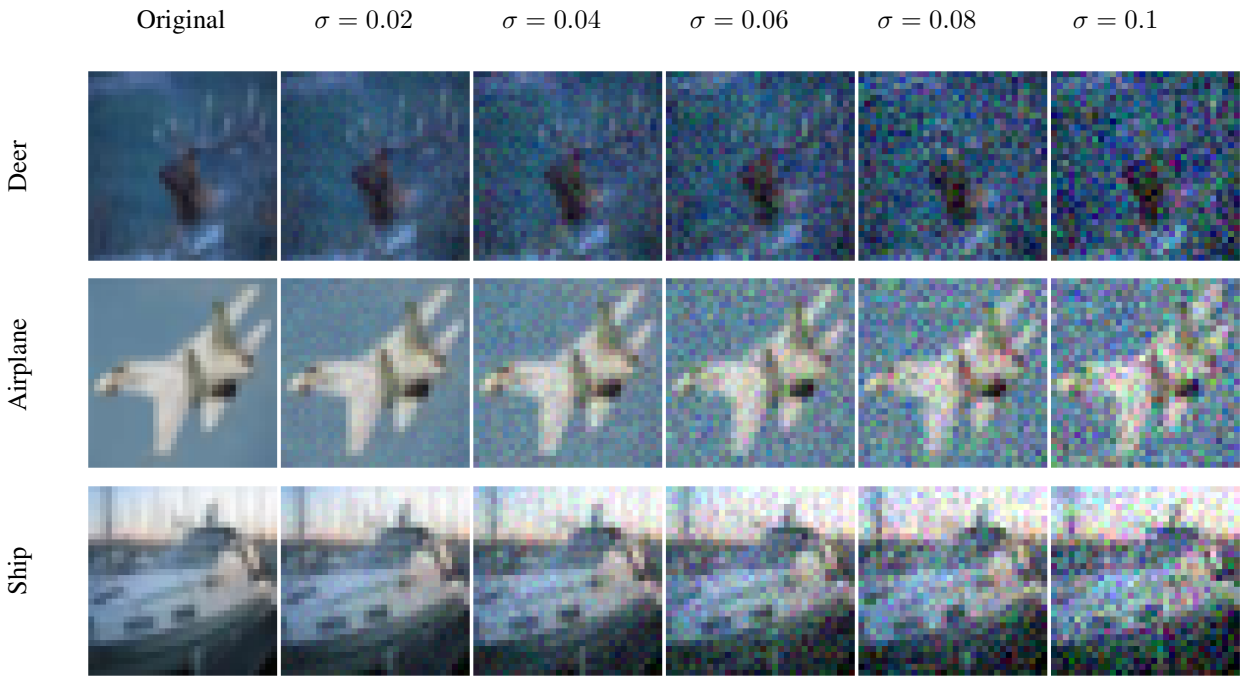


Figure 18: Examples of noisy images for CIFAR10.

		CPU: Intel x86_64		GPU: Tesla V100-PCIE-32GB	
Estimator	# samples (n)	Serial	Batched	Serial	Batched
p_σ^{mc}	$n = 100$	0:00:59	0:00:42	0:00:12	0:00:01
	$n = 1000$	0:09:50	0:07:22	0:02:00	0:00:04
	$n = 10000$	*1:41:11*	*1:14:38*	*0:19:56*	*0:00:35*
p_σ^{taylor}	N/A	0:00:08	0:00:07	0:00:02	$<$ 0:00:01
$p_\sigma^{\text{taylor\_mvs}}$	N/A	0:00:08	0:00:07	0:00:01	$<$ 0:00:01
p_σ^{mmse}	$n = 1$	0:00:08	0:00:10	0:00:02	0:00:02
	$n = 5$	*0:00:41*	*0:00:31*	*0:00:06*	*0:00:02*
	$n = 10$	0:01:21	0:01:02	0:00:11	0:00:02
	$n = 25$	0:03:21	0:02:44	0:00:26	0:00:03
	$n = 50$	0:06:47	0:05:38	0:00:51	0:00:04
	$n = 100$	0:13:57	0:11:31	0:01:42	0:00:06
$p_\sigma^{\text{mmse\_mvs}}$	$n = 1$	0:00:08	0:00:08	0:00:01	0:00:01
	$n = 5$	*0:00:41*	*0:00:32*	*0:00:05*	*0:00:01*
	$n = 10$	0:01:21	0:01:00	0:00:10	0:00:02
	$n = 25$	0:03:24	0:02:37	0:00:25	0:00:02
	$n = 50$	0:06:47	0:05:35	0:00:51	0:00:03
	$n = 100$	0:13:28	0:11:32	0:01:42	0:00:06
p_T^{softmax}	N/A	0:00:01	$<$ 0:00:01	$<$ 0:00:01	$<$ 0:00:01

Table 3: Runtimes of each p_σ^{robust} estimator. Each estimator computes $p_{\sigma=0.1}^{\text{robust}}$ for the CIFAR10 ResNet18 model for 50 data points. For estimators that use sampling, the row with the minimum number of samples necessary for convergence is italicized. Runtimes are in the format of hour:minute:second. The analytical estimators (p_σ^{taylor}, $p_\sigma^{\text{taylor\_mvs}}$, p_σ^{mmse}, and $p_\sigma^{\text{mmse\_mvs}}$) are more efficient than the naïve estimator (p_σ^{mc}).

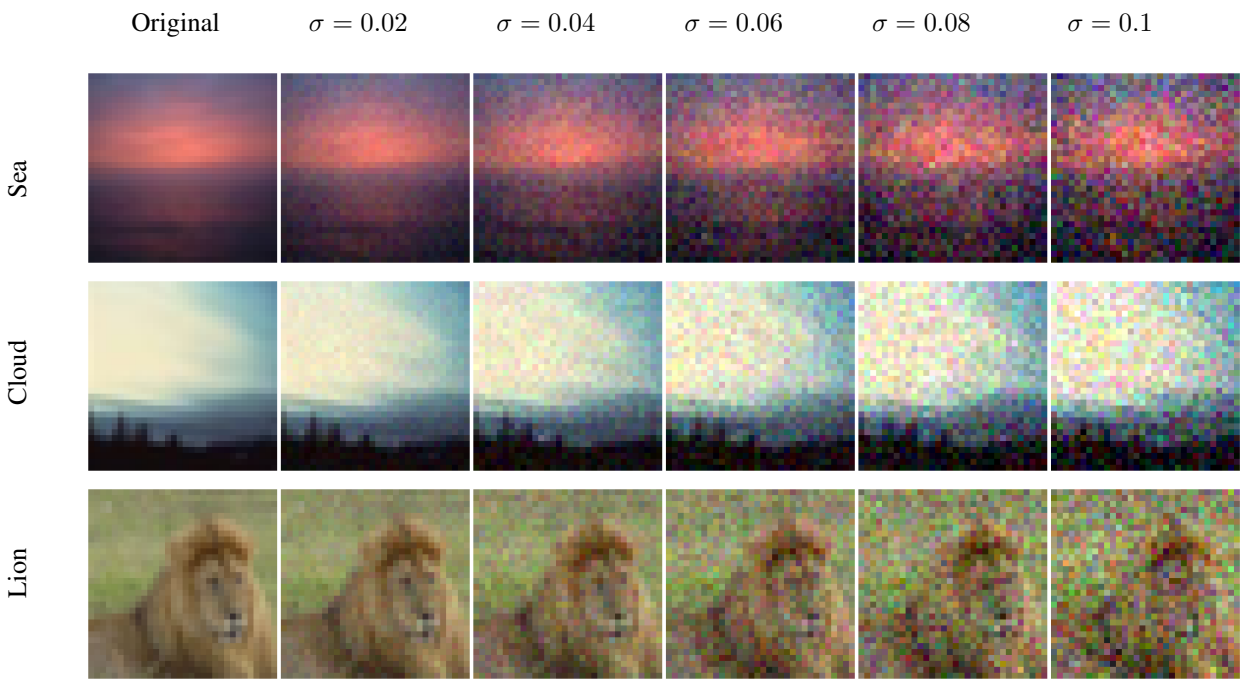


Figure 19: Examples of noisy images for CIFAR100.