Perturbation Type Categorization for Multiple Adversarial Perturbation Robustness: Supplementary Material

Pratyush Maini[1] **Xinyun Chen**[2] **Bo Li**[3] **Dawn Song**[2]

[1]Carnegie Mellon University
[2]University of California, Berkeley
[3]University of Illinois at Urbana-Champaign

A PROBLEM SETTING: THEORETICAL ANALYSIS

The classification problem consists of two tasks: **(1)** Predicting the correct class label of an adversarially perturbed (or benign) image using adversarially robust classifier $M_{\mathcal{A}}$; and **(2)** Predicting the type of adversarial perturbation that the input image was subjected to, using attack classifier C_{adv}.

Setup. We consider the data to consist of inputs to be sampled from two multi-variate Gaussian distributions such that the input-label pairs (x,y) can be described as:

$$\begin{aligned} y &\overset{u.a.r}{\sim} \{-1,+1\}, \\ x_0 \sim \mathcal{N}(y\alpha,\sigma^2), \quad x_1,\ldots,x_d &\overset{i.i.d}{\sim} \mathcal{N}(y\eta,\sigma^2), \end{aligned} \tag{1}$$

where the input $x \sim \mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma}) \in \mathcal{R}^{(d+1)}$; $\eta = \alpha/\sqrt{d}$ for some positive constant α; $\boldsymbol{\mu} = [\alpha,\eta,\ldots,\eta] \in \mathcal{R}^{+(d+1)}$ and $\boldsymbol{\Sigma} = \sigma^2\mathbf{I} \in \mathcal{R}^{+(d+1)\times(d+1)}$. We can assume without loss of generality, that the mean for the two distributions has the same absolute value, since for any two distributions with mean $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$, we can translate the origin to $\frac{\boldsymbol{\mu}_1+\boldsymbol{\mu}_2}{2}$. This setting demonstrates the distinction between an input feature x_0 that is strongly correlated with the input label and d weakly correlated features that are normally distributed (independently) with mean $y\eta$ and variance σ^2 each. We adapt this setting from Ilyas et al. [2019] who used a stochastic feature $x_0 = y$ with probability p, as opposed to a normally distributed input feature as in our case. All our findings hold in the other setting as well, however, the chosen setting better represents true data distribution, with some features that are strongly correlated to the input label, while others that have only a weak correlation.

B SEPARABILITY OF PERTURBATION TYPES (THEOREM 1)

Our goal is to evaluate if the optimal perturbation confined within different ℓ_p balls have different distributions and whether they are separable. We do so by developing an error bound on the maximum error in classification of the perturbation types. The goal of the adversary is to fool a standard (non-robust) classifier M. C_{adv} aims to predict the perturbation type based on **only** viewing the adversarial image, and not the delta perturbation.

First, in Appendix B.1 we define a binary Gaussian classifier that is trained on the given task. Given the weights of the binary classifier, we then identify the optimal adversarial perturbation for each of the $\ell_1, \ell_2, \ell_\infty$ attack types in Appendix B.2. In Appendix B.3 we define the difference between the adversarial input distribution for different ℓ_p balls. Finally, we calculate the error in classification of these adversarial input types in Appendix B.4 to conclude the proof of Theorem 1.

B.1 BINARY GAUSSIAN CLASSIFIER

We assume that we have enough input data to be able to empirically estimate the parameters μ, σ of the input distribution via sustained sampling. The multivariate Gaussian representing the input data is given by:

Accepted for the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022).

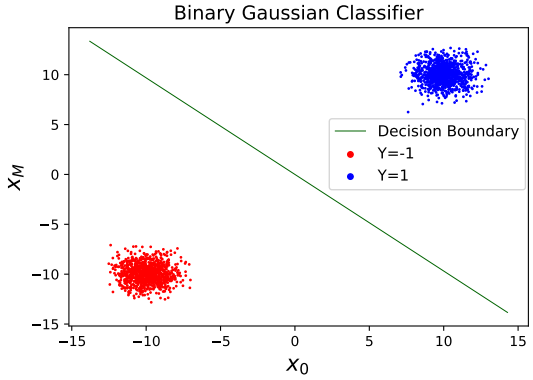


Figure 1: **Simulation**: Decision boundary (solid green line) of binary Gaussian classifier. $x_M = \frac{1}{\sqrt{d}} \sum_{i=1}^{d} x_i$ represents a meta feature, and x_0 is the first dimension of the input.

$$p(x|y = y_i) = \frac{1}{\sqrt{(2\pi)^d |\mathbf{\Sigma}|}} \exp\left(-\frac{1}{2}(x - y_i.\boldsymbol{\mu})^T \mathbf{\Sigma}^{-1}(x - y_i.\boldsymbol{\mu})\right), \quad \forall y_i \in \{-1, 1\}. \tag{2}$$

We want to find $p(y = y_i|x)\ \forall y_i \in \{-1, +1\}$. From Bayesian Decision Theory, the optimal decision rule for separating the two distributions is given by:

$$\begin{aligned} p(y = 1)p(x|y = 1) &\overset{y=1}{>} p(y = -1)p(x|y = -1); \\ p(y = 1)p(x|y = 1) &\overset{y=-1}{<} p(y = -1)p(x|y = -1). \end{aligned} \tag{3}$$

Therefore, for two Gaussian Distributions $\mathcal{N}(\boldsymbol{\mu}_1, \mathbf{\Sigma}_1), \mathcal{N}(\boldsymbol{\mu}_2, \mathbf{\Sigma}_2)$, we have:

$$\begin{aligned} &0 \overset{y=1}{<} x^\top A x - 2b^\top x + c; \\ &A = \mathbf{\Sigma}_1^{-1} - \mathbf{\Sigma}_2^{-1}; \\ &b = \mathbf{\Sigma}_1^{-1}\mu_1 - \mathbf{\Sigma}_2^{-1}\mu_2; \\ &c = \mu_1^\top \mathbf{\Sigma}_1^{-1}\mu_1 - \mu_2^\top \mathbf{\Sigma}_2^{-1}\mu_2 + \log\frac{\|\Sigma_1\|}{\|\Sigma_2\|} - 2\log\frac{p(y=1)}{p(y=-1)}. \end{aligned} \tag{4}$$

Substituting (2) and (3) in (4), we find that the optimal Bayesian decision rule for our problem is given by:

$$x^\top \boldsymbol{\mu} \overset{y=1}{>} 0, \tag{5}$$

which means that the label for the input can be predicted with the information of the sign of $x^\top \boldsymbol{\mu}$ alone. We can define the parameters $\mathbf{W} \in \mathcal{R}^{d+1}$ of the optimal binary Gaussian classifier M^W , such that $\|\mathbf{W}\|_2 = 1$ as:

$$\begin{aligned} \mathbf{W}_0 = \frac{\alpha}{\sqrt{2}}, \qquad \mathbf{W}_i = \frac{\alpha}{\sqrt{2d}} \quad \forall i \in \{1, \ldots, d\}; \\ M^W(x) = x^\top W. \end{aligned} \tag{6}$$

The same is also verified via a simulation in Figure 1.

B.2 OPTIMAL ADVERSARIAL PERTURBATION AGAINST M^W

Now, we calculate the optimal perturbation δ that is added to an input by an adversary in order to fool our model. For the purpose of this analysis, we only aim to fool a model trained on the standard classification metric as discussed in Section 3 (and not an adversarially robust model). The parameters of our model are defined in (6).

The objective of any adversary $\delta \in \Delta$ is to maximize the loss of the label classifier M^W. We assume that the classification loss is given by $-y \times M^W(x+\delta)$. The object of the adversary is to find δ^* such that:

$$\begin{aligned} \ell(x+\delta, y; M^W) &= -y \times M^W(x+\delta) = -yx^\top \mathbf{W}; \\ \delta^* &= \arg\max_{\delta \in \Delta} \ell(x+\delta, y; M^W), \\ &= \arg\max_{\delta \in \Delta} -y(x+\delta)^\top \mathbf{W} = \arg\max_{\delta \in \Delta} -y\delta^\top \mathbf{W}. \end{aligned} \tag{7}$$

We will now calculate the optimal perturbation in the ℓ_p balls $\forall p \in \{1, 2, \infty\}$. For the following analyses, we restrict the perturbation region Δ to the corresponding ℓ_p ball of radius $\{\epsilon_1, \epsilon_2, \epsilon_\infty\}$ respectively. We also note that the optimal perturbation exists at the boundary of the respective ℓ_p balls. Therefore, the constraint can be re-written as :

$$\delta^* = \arg\max_{\|\delta\|_p = \epsilon_p} -y\delta^\top \mathbf{W}. \tag{8}$$

We use the following properties in the individual treatment of ℓ_p balls:

$$\begin{aligned} \|\delta\|_p &= \left(\sum_i |\delta_i|^p \right)^{\frac{1}{p}}, \\ \partial_j \|\delta\|_p &= \frac{1}{p} \left(\sum_i |\delta_i|^p \right)^{\frac{1}{p}-1} \cdot p|\delta_j|^{p-1} \operatorname{sgn}(\delta_j) = \left(\frac{|\delta_j|}{\|\delta\|_p} \right)^{p-1} \operatorname{sgn}(\delta_j). \end{aligned} \tag{9}$$

p = 2 Making use of langrange multipliers to solve (8), we have:

$$\begin{aligned} \nabla_\delta(-\delta^\top \Sigma^{-1}\mu) &= \lambda \nabla_\delta(\|\delta\|_p^2 - \epsilon_p^2), \\ -\mathbf{W} &= \lambda^{'} \|\delta\|_p \nabla_\delta(\|\delta\|_p). \end{aligned} \tag{10}$$

Combining the results from (9) and replacing δ with δ_2 we obtain :

$$\begin{aligned} -\mathbf{W} &= \lambda^{'} \|\delta_2\|_2 \left(\frac{|\delta_2|}{\|\delta_2\|_2} \right) \operatorname{sgn}(\delta_2) \\ \delta_2; &= -\epsilon_2 \left(\frac{\mathbf{W}}{\|\mathbf{W}\|_2} \right) = -\epsilon_2 \mathbf{W}. \end{aligned} \tag{11}$$

p = ∞ Recall that the optimal perturbation is given by :

$$\begin{aligned} \delta^* &= \arg\max_{\|\delta\|_\infty = \epsilon_\infty} -y\delta^\top \mathbf{W}, \\ &= \arg\max_{\|\delta\|_\infty = \epsilon_\infty} -y \sum_{i=0}^{d} \delta_i \mathbf{W}_i. \end{aligned} \tag{12}$$

Since $\|\delta\|_\infty = \epsilon_\infty$, we know that $\max_i |\delta_i| = \epsilon_\infty$. Therefore (12) is maximized when each $\delta_i = -y\epsilon_\infty \operatorname{sgn} \mathbf{W}_i \quad \forall i \in \{0, \ldots, d\}$. Further, since the weight matrix only contains non-negative elements (α is a positive constant), we can conclude that the optimal perturbation is given by:

$$\delta_\infty = -y\epsilon_\infty \mathbf{1}. \tag{13}$$

p = 1 We attempt an analytical solution for the optimal perturbation δ_1. Recall that the optimal perturbation is given by :

$$\begin{aligned}\delta^* &= \arg\max_{\|\delta\|_1=\epsilon_1} -y\sum_{i=1}^{d}\delta_i\mathbf{W}_i,\\ &= \arg\max_{\|\delta\|_1=\epsilon_1} -y\delta_0\mathbf{W}_0 - y\sum_{i=1}^{d}\delta_i\mathbf{W}_i,\\ &= \arg\max_{\|\delta\|_1=\epsilon_1} -y\delta_0\frac{\alpha}{\sqrt{2}} - y\sum_{i=1}^{d}\delta_i\frac{\alpha}{\sqrt{2d}}.\end{aligned} \tag{14}$$

Since $\|\delta\|_1 = \epsilon_1$, (14) is maximized when:

$$\delta_0 = -y\epsilon_1\,\mathrm{sgn}(\alpha) = -y\epsilon_1, \qquad \delta_i = 0 \quad \forall i \in \{1\dots d\}. \tag{15}$$

Combining the results. From the preceding discussion, it may be noted that the new distribution of inputs within a given label changes by a different amount δ depending on the perturbation type. Moreover, if the mean and variance of the distribution of a given label are known (which implies that the corresponding true data label is also known), the optimal perturbation is independent of the input itself, and only dependent on the respective class statistics (Note that the input is still important in order to understand the true class).

B.3 PERTURBATION CLASSIFICATION BY C_{adv}

Now we aim to verify if it is possible to accurately separate the optimal adversarial inputs crafted within different ℓ_p balls. For the purposes of this discussion, we only consider the problem of classifying perturbation types into ℓ_1 and ℓ_∞, but the same analysis may also be extended more generally to any number of perturbation types.

We will consider the problem of classifying the correct attack label for inputs from true class $y = 1$ for this discussion. Note that the original distribution:

$$X_{true} \sim \mathcal{N}(y.\boldsymbol{\mu},\, \boldsymbol{\Sigma}).$$

Since the perturbation value δ_p is fixed for all inputs corresponding to a particular label, the new distribution of perturbed inputs X_1 and X_∞ in case of ℓ_1 and ℓ_∞ attacks respectively (for y = 1) is given by:

$$\begin{aligned}X_1 &\sim \mathcal{N}(\boldsymbol{\mu} + \delta_1,\, \boldsymbol{\Sigma});\\ X_\infty &\sim \mathcal{N}(\boldsymbol{\mu} + \delta_\infty,\, \boldsymbol{\Sigma}).\end{aligned} \tag{16}$$

We now try to evaluate the conditions under which we can separate the two Gaussian distributions with an acceptable worst-case error.

B.4 CALCULATING A BOUND ON THE ERROR

Classification Error. A classification error occurs if a data vector x belongs to one class but falls in the decision region of the other class. That is in (3) the decision rule indicates the incorrect class. (This can be understood through the existence of outliers)

$$\begin{aligned}P_e &= \int P(\text{error}|x)p(x)dx,\\ &= \int \min\left[p(y=\ell_1|x)p(x), p(y=\ell_\infty|x)p(x)\right]dx.\end{aligned} \tag{17}$$

Perturbation Size. We set the radius of the ℓ_∞ ball, $\epsilon_\infty = \eta$ and the radius of the ℓ_1 ball, $\epsilon_1 = \alpha$. We further extend the discussion about suitable perturbation sizes in Appendix C.2. These values ensure that the ℓ_∞ adversary can make all the weakly correlated labels meaningless by changing the expected value of the adversarial input to less than 0 ($\mathbf{E}[x_i + \delta_\infty(i)] \quad \forall i > 0$), while the ℓ_1 adversary can make the strongly correlated feature x_0 meaningless by changing its expected value to less than 0 ($\mathbf{E}[x_0 + \delta_1(0)]$). However, neither of the two adversaries can flip all the features together.

Translating the axes. We can translate the axis of reference by $\left(-\mu - \left(\frac{\delta_1+\delta_\infty}{2}\right)\right)$ and define $\boldsymbol{\mu}_{adv} = \left(\frac{\delta_1-\delta_\infty}{2}\right)$, such that :

$$\begin{aligned} X_1 &\sim \mathcal{N}(\boldsymbol{\mu}_{adv}, \boldsymbol{\Sigma}); \\ X_\infty &\sim \mathcal{N}(-\boldsymbol{\mu}_{adv}, \boldsymbol{\Sigma}). \end{aligned} \tag{18}$$

We can once again combine this with the simplified Bayesian model in (5) to obtain the classification rule:

$$x^\top \boldsymbol{\mu}_{adv} \overset{p=1}{>} 0. \tag{19}$$

Combining the optimal perturbation definitions in (13) and (15) that $\boldsymbol{\mu}_{adv} = \left(\frac{\delta_1-\delta_\infty}{2}\right) = \frac{1}{2}\left[-\epsilon_1 + \epsilon_\infty, \epsilon_\infty, \ldots, \epsilon_\infty\right]$. We can further substitute $\epsilon_1 = \alpha$ and $\epsilon_\infty = \eta = \frac{\alpha}{\sqrt{d}}$. Notice that $\boldsymbol{\mu}_{adv}(i) > 0 \; \forall i > 0$. Without loss of generality, to simplify further discussion we can flip the coordinates of x_0, since all dimensions are independent of each other. Therefore, $\boldsymbol{\mu}_{adv} = \frac{\alpha}{2\sqrt{d}}\left[\sqrt{d} - 1, 1, \ldots, 1\right]$. Consider a new variable x_z such that:

$$x_z = x_0 \cdot \left(1 - \frac{1}{\sqrt{d}}\right) + \frac{1}{\sqrt{d}} \sum_{i=1}^{d} x_i = \frac{2}{\alpha}\left(x^\top \boldsymbol{\mu}_{adv}\right). \tag{20}$$

Since each $x_i \forall i \geq 0$ is independently distributed, the new feature $x_z \sim \mathcal{N}(\mu_z, \sigma_z^2)$, where

$$\begin{aligned} \mu_z &= \alpha\left(1 - \frac{1}{\sqrt{d}}\right) + \frac{1}{\sqrt{d}} \sum_{i=1}^{d} \frac{\alpha}{\sqrt{d}} = 2\alpha - \frac{\alpha}{\sqrt{d}} \\ \sigma_z^2 &= \sigma^2\left(1 + \frac{1}{d} - 2\frac{1}{\sqrt{d}} + \sum_{i=1}^{d} \frac{1}{d}\right), \\ &= \sigma^2\left(2 + \frac{1}{d} - 2\frac{1}{\sqrt{d}}\right). \end{aligned} \tag{21}$$

Therefore, the problem simplifies to calculating the probability that the meta-variable $x_z > 0$.

For $\frac{\alpha}{\sigma} > 10$ and $d > 1$, we have in the z-table, $z > 10$:

$$P_e \leq 10^{-24}, \tag{22}$$

which suggests that the distributions are significantly distinct and can be easily separated. This concludes the proof for Theorem 1.

Note: We can extend the analysis to other ℓ_p balls as well, but we consider ℓ_1 and ℓ_∞ for simplicity.

C ROBUSTNESS OF THE PROTECTOR PIPELINE (THEOREM 2)

In the previous section, we show that it is indeed possible to distinguish between the distribution of inputs of a given class that were subjected to ℓ_1 and ℓ_∞ perturbations over a standard classifier. Now, we aim to develop further understanding of the robustness of our two-stage pipeline in a dynamic attack setting with multiple labels to distinguish among. The first stage is a preliminary classifier C_{adv} that classifies the perturbation type and the second stage consists of multiple models $M_{\mathcal{A}}$ that were specifically trained to be robust to perturbations to the input within the corresponding ℓ_p norm.

First, in Appendix C.1, we calculate the optimal weights for a binary Gaussian classifier $M_{\mathcal{A}}$, trained on dataset $\mathcal{D}$ to be robust to adversaries within the ℓ_p ball $\forall p \in \{1, \infty\}$. Based on the weights of the individual model, we fix the perturbation size ϵ_p to be only as large, as is required to fool the alternate model with high probability. Here, by 'alternate' we mean that for an ℓ_q attack, the prediction should be made by the M_{ℓ_p, ϵ_p} model,where $p, q \in \{1, \infty\}; p \neq q$. In Appendix C.3 we

calculate the robustness of individual $M_{\mathcal{A}}$ models to ℓ_p adversaries, given the perturbation size ϵ_p as defined in Appendix C.2. In Appendix C.4, we analyze the modified distributions of the perturbed inputs after different ℓ_p attacks. Based on this analysis, we construct a simple decision rule for the perturbation classifier C_{adv}. Finally, in Appendix C.5 we determine the perturbation induced by the worst-case adversary that has complete knowledge of both C_{adv} and $M_{\ell_p,\epsilon_p} \forall p \in \{1, \infty\}$. We show how there exists a trade-off between fooling the perturbation classifier (to allow the alternate M_{ℓ_p,ϵ_p} model to make the final prediction), and fooling the alternate M_{ℓ_p,ϵ_p} model itself.

Perturbation Size. We set the radius of the ℓ_∞ ball, $\epsilon_\infty = \eta + \zeta_\infty$ and the radius of the ℓ_1 ball, $\epsilon_1 = \alpha + \zeta_1$, where ζ_p are some small positive constants that we calculate in Appendix C.2. These values ensure that the ℓ_∞ adversary can make all the weakly correlated labels meaningless by changing the expected value of the adversarial input to less than 0 ($\mathbf{E}[x_i + \delta_\infty(i)] \quad \forall i > 0$), while the ℓ_1 adversary can make the strongly correlated feature x_0 meaningless by changing its expected value to less than 0 ($\mathbf{E}[x_0 + \delta_1(0)]$). However, neither of the two adversaries can flip all the features together. The exact values of ζ_p determine the exact success probability of the attacks. We defer this calculation to later when we have calculated the weights of the models $M_{\mathcal{A}}$. For the following discussion, it may be assumed that $\zeta_p \to 0 \; \forall p \in \{1, \infty\}$.

C.1 BINARY GAUSSIAN CLASSIFIER $M_{\mathcal{A}}$

Extending the discussion in Appendix B.1, we now examine the learned weights of a binary Gaussian classifier $M_{\mathcal{A}}$ that is trained to be robust against perturbations within the corresponding ℓ_p ball of radius ϵ_p. The optimization equation for the classifier can be formulated as follows:

$$\min_{\mathbf{W}} \mathbb{E}\left[-yx^\top \mathbf{W}\right] + \frac{1}{2}\lambda ||\mathbf{W}||_2^2, \tag{23}$$

where λ is tuned in order to make the ℓ_2 norm of the optimal weight distribution, $||\mathbf{W}^*||_2, = 1$. Following the symmetry argument in Lemma D.1 [Tsipras et al., 2018] we extend for the binary Gaussian classifier that :

$$\mathbf{W}_i^* = \mathbf{W}_j^* = \mathbf{W_M} \quad \forall i, j \in \{1, \ldots, d\}. \tag{24}$$

We deal with the cases pertaining to $p \in \{\infty, 1\}$ in this section. For both the cases, we consider existential solutions for the classifier $M_{\mathcal{A}}$ to simplify the discussion. This gives us lower bounds on the performance of the optimal robust classifier. The robust objective under adversarial training can be defined as:

$$\begin{aligned} &\min_{\mathbf{W}} \max_{\|\delta\|_p \le \epsilon_p} \mathbb{E}\left[\mathbf{W}_0 \cdot (x_0 + \delta_0) + \mathbf{W_M} \cdot \sum_{i=1}^{d} (x_i + \delta_i)\right] + \frac{1}{2}\lambda \|\mathbf{W}\|_2^2; \\ &\min_{\mathbf{W}} \left\{-1\left(\mathbf{W_0}\alpha + d \times \mathbf{W_M}\frac{\alpha}{\sqrt{d}}\right) + \frac{1}{2}\lambda\|\mathbf{W}\|_2^2 + \max_{\|\delta\|_p \le \epsilon_p} \mathbb{E}\left[-y\left(\mathbf{W}_0\delta_0 + \mathbf{W_M}\sum_{i=1}^{d}\delta_i\right)\right]\right\} \end{aligned} \tag{25}$$

Further, since the λ constraint only ensures that $||\mathbf{W}^*||_2 = 1$, we can simplify the optimization equation by substituting $\mathbf{W_0} = \sqrt{1 - d \cdot \mathbf{W_M}^2}$ as follows,

$$\min_{\mathbf{W_M}} \left\{-1\left(\alpha\sqrt{1 - d \cdot \mathbf{W_M}^2} + d \times \mathbf{W_M}\frac{\alpha}{\sqrt{d}}\right) + \max_{\|\delta\|_p \le \epsilon_p} \mathbb{E}\left[-y\left(\delta_0\sqrt{1 - d \cdot \mathbf{W_M}^2} + \mathbf{W_M}\sum_{i=1}^{d}\delta_i\right)\right]\right\}. \tag{26}$$

p = ∞ As discussed in (13) the optimal perturbation δ_∞ is given by $-y\epsilon_\infty \mathbf{1}$. The optimization equation is simplified to:

$$\min_{\mathbf{W_M}} \left\{(\epsilon_\infty - \alpha)\sqrt{1 - d \cdot \mathbf{W_M}^2} + d \times \mathbf{W_M}\left(\epsilon_\infty - \frac{\alpha}{\sqrt{d}}\right)\right\}. \tag{27}$$

Recall that $\epsilon_\infty = \frac{\alpha}{\sqrt{d}} + \zeta_\infty$. To simplify the following discussion we use the weights of a classifier trained to be robust against perturbations within the ℓ_∞ ball of radius $\epsilon_\infty = \frac{\alpha}{\sqrt{d}}$. The optimal solution is then given by:

$$\lim_{\zeta_\infty \to 0} \mathbf{W_M} = 0. \tag{28}$$

Therefore, the classifier weights are given by $\mathbf{W} = [\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_d] = [1, 0, \ldots, 0]$. We also show later in Appendix C.3 that the model achieves greater than 99% accuracy against ℓ_∞ adversaries for the chosen values of ζ_∞.

p = 1 We consider an analytical solution to yield optimal weights for this case. Recall from (15) that the optimal perturbation δ_1 depends on the weight distribution of the classifier. Therefore, if $\mathbf{W}_0 > \mathbf{W_M}$ the optimization equation can be simplified to

$$\min_{\mathbf{W}} \left\{ \mathbf{W_0}(\epsilon_1 - \alpha) - d \times \mathbf{W_M} \frac{\alpha}{\sqrt{d}} + \frac{1}{2}\lambda \|\mathbf{W}\|_2^2 \right\}, \tag{29}$$

and if $\mathbf{W_M} > \mathbf{W}_0$

$$\min_{\mathbf{W}} \left\{ -\mathbf{W_0}\alpha - \mathbf{W_M}\left(\sqrt{d}\alpha - \epsilon_1\right) + \frac{1}{2}\lambda \|\mathbf{W}\|_2^2 \right\}. \tag{30}$$

Recall that $\epsilon_1 = \alpha + \zeta_1$. Once again to simplify the discussion that follows we will lower bound the robust accuracy of the classifier M_{ℓ_1} by considering the optimal solution when $zeta_1 = 0$. The optimal solution is then given by:

$$\lim_{\zeta_1 \to 0} \mathbf{W_M} = 1. \tag{31}$$

For the robust classifier M_{ℓ_1}, the weights $\mathbf{W} = [\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_d] = [0, \frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}, \ldots, \frac{1}{\sqrt{d}}]$. While this may not be the optimal solution for all values of ζ_1, we are only interested in a lower bound on the final accuracy and the classifier described by weights $\mathbf{W}$ simplifies the discussion hereon. We also show later in Appendix C.3 that the model achieves greater than 99% accuracy against ℓ_1 adversaries for the chosen values of ζ_1.

C.2 PERTURBATION SIZES FOR FOOLING $M_{\mathcal{A}}$ MODELS

Now that we exactly know the weights of the learned robust classifiers M_{ℓ_1} and M_{ℓ_∞}, we can move towards calculating values ζ_1 and ζ_∞ for the exact radius of the perturbation regions for the ℓ_1 and ℓ_∞ metrics. We set the radii of these regions in such a way that an ℓ_1 adversary can fool the model M_{ℓ_∞} with probability $\sim 98\%$ (corresponding to $z = 2$ in the z-table for normal distributions), and similarly, the success of ℓ_∞ attacks against the M_{ℓ_1} model is $\sim 98\%$.

Let P_{p_1,p_2} represent the probability that model $M_{\ell_{p_1}}$ correctly classifies an adversarial input in the ℓ_{p_2} region. For $p_1 = \infty$ and $p_2 = 1$,

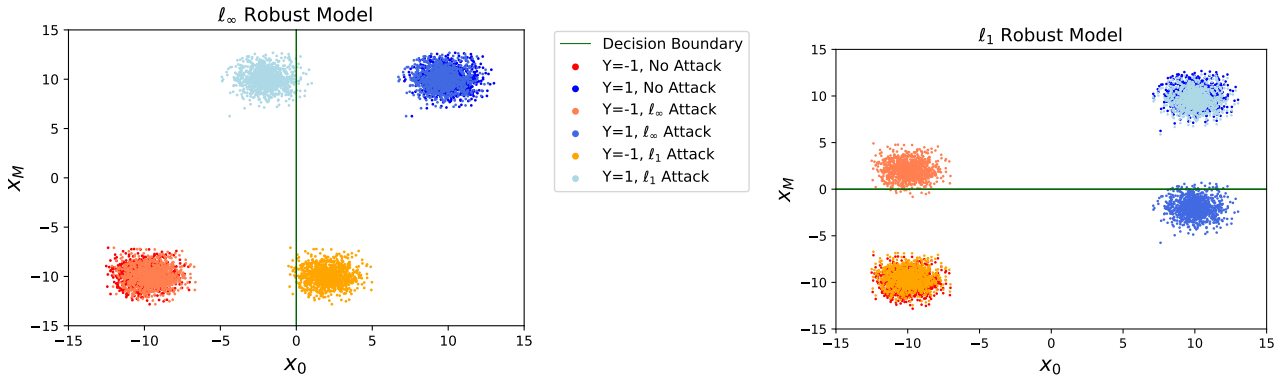
$$\begin{aligned}
P_{\infty,1} &= \mathbb{P}_{x \sim \mathcal{N}(y\boldsymbol{\mu}, \boldsymbol{\Sigma})}[y \cdot M_{\ell_\infty}(x + \delta_1) > 0], \\
&= \mathbb{P}_{x \sim \mathcal{N}(y\boldsymbol{\mu}, \boldsymbol{\Sigma})}[y \cdot (x + \delta_1)^\top \mathbf{W} > 0], \\
&\geq \mathbb{P}_{x \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})}[x_0 > \epsilon_1]; \\
z &= \frac{\epsilon_1 - \alpha}{\sigma} = \frac{\alpha + \zeta_1 - \alpha}{\sigma} = \frac{\zeta_1}{\sigma} = 2; \\
\zeta_1 &= 2\sigma; \\
\epsilon_1 &= \alpha + 2\sigma.
\end{aligned} \tag{32}$$

To simplify the discussion for the M_{ℓ_1} model, we define a meta-feature x_M as:

$$x_M = \frac{1}{\sqrt{d}} \sum_{i=1}^{d} x_i, \tag{33}$$

which is distributed as :

$$x_M \sim \mathcal{N}(y\eta\sqrt{d}, \sigma^2) \stackrel{d}{=} \mathcal{N}(y\alpha, \sigma^2).$$



(a) (b)

Figure 2: **Simulation:** Decision boundary (solid green line) and robustness of individual $M_{\mathcal{A}}$ models to different ℓ_p attacks. x_M represents the meta feature as defined in Equation 33 and x_0 is the first dimension of the input. Notice how the distribution of perturbed samples varies according to the change in model architecture (scatter plots in the same color in the two graphs represent the same distribution). (a) The M_{ℓ_∞} model is able to correctly classify all benign and ℓ_∞ perturbed samples. However, the ℓ_1 adversary is able to successfully flip the decision of most data points (b) The same illustration is repeated for the M_1 model. In this case, while the model is robust to ℓ_1 attacks, it fails against an ℓ_∞ adversary.

For $p_1 = 1$ and $p_2 = \infty$,

$$
\begin{aligned}
P_{1,\infty} &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot M_{\ell_1}(x+\delta_\infty) > 0], \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x+\delta_\infty)^\top \mathbf{W} > 0], \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot \frac{1}{\sqrt{d}} \sum_{i=1}^{d} (x_i + \delta_\infty(i)) > 0], \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x_M - \sqrt{d}\cdot\epsilon_\infty) > 0], \\
&\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[x_M > \sqrt{d}\cdot\epsilon_\infty\right]; \\
z &= \frac{\sqrt{d}\cdot\epsilon_\infty - \alpha}{\sigma} = \frac{\alpha + \sqrt{d}\cdot\zeta_\infty - \alpha}{\sigma} = \frac{\sqrt{d}\cdot\zeta_\infty}{\sigma} = 2; \\
\zeta_\infty &= \frac{2\sigma}{\sqrt{d}}; \\
\epsilon_\infty &= \frac{\alpha + 2\sigma}{\sqrt{d}};
\end{aligned}
\tag{34}
$$

C.3 ROBUSTNESS OF INDIVIDUAL $M_{\mathcal{A}}$ MODELS

Additional assumptions. We add the following assumptions: (1) the dimensionality parameter d of input data is larger than 100; and (2) the ratio of the mean and variance for feature x_0 is greater than 10. (These assumptions were also made when introducing the problem in the main paper.)

$$
d \geq 100, \qquad \frac{\alpha}{\sigma} \geq 10. \tag{35}
$$

We define P_p as the probability that for any given input $x \sim \mathcal{N}(y\boldsymbol{\mu}, \boldsymbol{\Sigma})$, the classifier $M_{\mathcal{A}}$ outputs the correct label y for the input $x + \delta_p$.

p = ∞

$$\begin{aligned} P_{\infty,\infty} &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot M_{\ell_\infty}(x+\delta_\infty) > 0], \\ &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x+\delta_\infty)^\top \mathbf{W} > 0], \\ &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x_0 + \delta_\infty(0)) > 0], \\ &\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}[x_0 > \epsilon_\infty]; \\ z = \frac{\epsilon_\infty - \alpha}{\sigma} &= \frac{\alpha}{\sigma}\left(\frac{1}{\sqrt{d}} - 1\right) + \frac{2}{\sqrt{d}}. \end{aligned} \tag{36}$$

using the assumptions in (35),

$$P_{\infty,\infty} \geq 0.999. \tag{37}$$

p = 1

$$\begin{aligned} P_{1,1} &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot M_{\ell_1}(x+\delta_1) > 0], \\ &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x+\delta_1)^\top \mathbf{W} > 0], \\ &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot \frac{1}{\sqrt{d}}\sum_{i=1}^{d}(x_i + \delta_1(i)) > 0], \\ &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x_M + \delta_M) > 0], \\ &\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[x_M > \frac{\epsilon_1}{\sqrt{d}}\right]; \\ z = \frac{\frac{\epsilon_1}{\sqrt{d}} - \alpha}{\sigma} &= \frac{\alpha}{\sigma}\left(\frac{1}{\sqrt{d}} - 1\right) + \frac{2}{\sqrt{d}}. \end{aligned} \tag{38}$$

using the assumptions in (35),

$$P_{1,1} \geq 0.999. \tag{39}$$

C.4 DECISION RULE FOR C_{adv}

We aim to provide a lower bound on the worst-case accuracy of the entire pipeline, through the existence of a simple decision tree C_{adv}. For given perturbation budgets ϵ_1 and ϵ_∞, we aim to understand the range of values that can be taken by the adversarial input. Consider the scenarios described in Table 1 below. The same is also corroborated via the empirical experiments shown in Figure 2.

Table 1: The table shows the range of the values that the mean can take depending on the decision taken by the adversary. μ_0^{adv} and μ_M^{adv} represent the new mean of the distribution of features x_0 and x_M after the adversarial perturbation.

Attack Type	μ_0^{adv}		μ_M^{adv}	
	y = 1	y = -1	y = 1	y = -1
None	α	$-\alpha$	$\eta\sqrt{d}$	$-\eta\sqrt{d}$
ℓ_∞	$\{\alpha-\epsilon_\infty, \alpha+\epsilon_\infty\}$	$\{-\alpha-\epsilon_\infty, -\alpha+\epsilon_\infty\}$	$\{\eta\sqrt{d}+\epsilon_\infty\sqrt{d}, \eta\sqrt{d}-\epsilon_\infty\sqrt{d}\}$	$\{-\eta\sqrt{d}+\epsilon_\infty\sqrt{d}, -\eta\sqrt{d}-\epsilon_\infty\sqrt{d}\}$
ℓ_1	$\{\alpha-\epsilon_1, \alpha+\epsilon_1\}$	$\{-\alpha-\epsilon_1, -\alpha+\epsilon_1\}$	$\{\eta\sqrt{d}+\epsilon_1/\sqrt{d}, \eta\sqrt{d}-\epsilon_1/\sqrt{d}\}$	$\{-\eta\sqrt{d}+\epsilon_1/\sqrt{d}, -\eta\sqrt{d}-\epsilon_1/\sqrt{d}\}$

Note that any adversary that moves the perturbation away from the y-axis is uninteresting for our comparison, since irrespective of a correct perturbation type prediction by C_{adv}, either of the two second level models naturally obtain a high accuracy on such inputs. Hence, we define the following decision rule with all the remaining cases mapped to ℓ_1 perturbation type.

$$C_{adv}(x) = \begin{cases} 1, & \text{if} \quad ||x_0| - \alpha| < \epsilon_\infty + \frac{\alpha}{2} \\ 0, & \text{otherwise} \end{cases} \tag{40}$$

where the output 1 corresponds to the classifier predicting the presence of ℓ_∞ perturbation in the input, while an output of 0 suggests that the classifier predicts the input to contain perturbations of the ℓ_1 type.

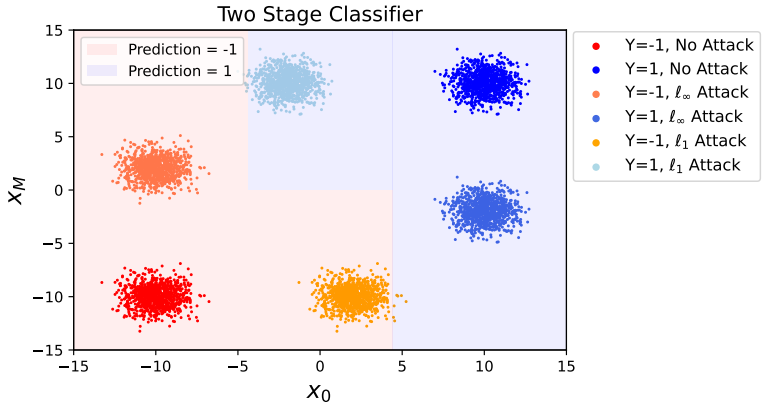


Figure 3: **Simulation**: Decision boundary of the overall two stage classifier. x_M represents the meta feature as defined in Equation 33 and x_0 is the first dimension of the input.

If we consider a black-box setting where the adversary has no knowledge of the classifier C_{adv}, and can only attack $M_{\mathcal{A}}$ it is easy to see that the proposed pipeline obtains a high adversarial accuracy against the union of ℓ_1 and ℓ_∞ perturbations (since the given decision rule correctly classifies known examples as simulated in Figure 2.

Note: (1) There exists a single model that can also achieve robustness against the union of ℓ_1 and ℓ_∞ perturbations, however, learning this model may be more challenging in real data settings. (2) The classifier need not be perfect.

C.5 TRADE-OFF BETWEEN ATTACKING $M_{\mathcal{A}}$ AND C_{adv}

To obtain true robustness it is important that the entire pipeline is robust against adversarial attacks. More specifically, in this section we demonstrate the natural tension that exists between fooling the top level attack classifier (by making an adversarial attack less representative of its natural distribution) and fooling the bottom level adversarially robust models (requiring stronger attacks leading to a return to the attack's natural distribution).

The accuracy of the pipelined model f against any input-label pair (x, y) sampled through some distribution $\mathcal{N}(y\boldsymbol{\mu}_{adv}, \boldsymbol{\Sigma})$ (where $\boldsymbol{\mu}_{adv}$ incorporates the change in the input distribution owing to the adversarial perturbation) is given by:

$$\begin{aligned}
\mathbb{P}\left[f(x) = y\right] &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right]\mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[y\cdot M_{\ell_\infty}(x) > 0 | C_{adv}(x)\right] \\
&\quad + (1 - \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right])\mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[y\cdot M_{\ell_1}(x) > 0 | \neg C_{adv}(x)\right], \\
&= \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right]\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_{\ell_\infty}(x) > 0 | C_{adv}(x)\right] \\
&\quad + (1 - \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right])\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_{\ell_1}(x) > 0 | \neg C_{adv}(x)\right].
\end{aligned} \tag{41}$$

ℓ_∞ adversary. To simplify the analysis, we consider loose lower bounds on the accuracy of the model f against the ℓ_∞ adversary. Recall that the decision of the attack classifier is only dependent of the input x_0. Irrespective of the input features $x_i \forall i > 0$, it is always beneficial for the adversary to perturb the input by $\mu_i = -\epsilon_\infty$. However, the same does not apply for the input x_0. Analyzing for the scenario when the true label $y = 1$, if the input x_0 lies between $\frac{\alpha}{2} + \epsilon_\infty$ of the mean α, irrespective of the perturbation, the output of the attack classifier $C_{adv} = 1$. The M_{ℓ_∞} model then always correctly classifies these inputs. The overall robustness of the pipeline requires analysis for the case when input lies outside $\frac{\alpha}{2} + \epsilon_\infty$ of the mean as well. However, we consider that the adversary always succeeds in such a case in order to only obtain a loose lower bound on the robust accuracy of the pipeline model f against ℓ_∞ attacks.

$$
\begin{aligned}
\mathbb{P}\left[f(x)=y\right] &= \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right]\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_{\ell_\infty}(x)>0|C_{adv}(x)\right], \\
&\qquad + (1-\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right])\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_{\ell_1}(x)>0|\neg C_{adv}(x)\right], \\
&\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right]\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_{\ell_\infty}(x)>0|C_{adv}(x)\right], \\
&\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[|x_0-\alpha|\leq\frac{\alpha}{2}-\epsilon_\infty\right], \\
&\geq 2\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[x_0\leq\alpha-\frac{\alpha}{2}+\epsilon_\infty\right], \\
z &= \frac{(\alpha-\frac{\alpha}{2}+\epsilon_\infty)-\alpha}{\sigma} = -\frac{\alpha}{2\sigma}+\frac{3\sigma}{2\sigma\sqrt{d}}.
\end{aligned}
\tag{42}
$$

using the assumptions in (35),

$$
\mathbb{P}\left[f(x)=y\right]\sim 0.99. \tag{43}
$$

ℓ_1 **adversary.** It may be noted that a trivial way for the ℓ_1 adversary to fool the attack classifier is to return a perturbation $\delta_1 = 0$. In such a scenario, the classifier predicts that the adversarial image was subjected to an ℓ_∞ attack. The label prediction is hence made by the M_{ℓ_∞} model. But we know from (37) that the M_{ℓ_∞} model predicts benign inputs correctly with a probability $P_{\infty,\infty} > 0.99$, hence defeating the adversarial objective of misclassification. To achieve misclassification over the entire pipeline the optimal perturbation decision for the ℓ_1 adversary when $x_0 \in \left[-\alpha-\frac{\alpha}{2}-\epsilon_1, -\alpha+\frac{\alpha}{2}+\epsilon_1\right]$ the adversary can fool the pipeline by ensuring that the $C_{adv}(x) = 1$. However, in all the other cases irrespective of the perturbation, either $C_{adv} = 0$ or the input features x_0 has the same sign as the label y. Since, $P_{1,1} > 0.99$ for the M_{ℓ_1} model, for all the remaining inputs x_0 the model correctly predicts the label with probability greater than 0.99 (approximate lower bound). We formulate this trade-off to elaborate upon the robustness of the proposed pipeline.

$$
\begin{aligned}
\mathbb{P}\left[f(x)=y\right] &= \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right]\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_{\ell_\infty}(x)>0|C_{adv}(x)\right] \\
&\qquad + (1-\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right])\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_{\ell_1}(x)>0|\neg C_{adv}(x)\right], \\
&\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[-\alpha-\frac{\alpha}{2}-\epsilon_1\leq x_0\leq-\alpha+\frac{\alpha}{2}+\epsilon_1\right] \\
&\qquad + 0.999(\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[x_0<-\alpha-\frac{\alpha}{2}-\epsilon_1 \text{ or } x_0>-\alpha+\frac{\alpha}{2}+\epsilon_1\right]), \\
&\geq 0.999(\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[x_0<-\alpha-\frac{\alpha}{2}-\epsilon_1 \text{ or } x_0>-\alpha+\frac{\alpha}{2}+\epsilon_1\right]).
\end{aligned}
\tag{44}
$$

using the assumptions in (35),

$$
\mathbb{P}\left[f(x)=y\right]\sim 0.99. \tag{45}
$$

This concludes the proof for Theorem 2, showing that an adversary can hardly stage successful attacks on the entire pipeline and faces a natural tension between attacking the label predictor and the attack classifier. We verify these results via a simulation in Figure 3. We emphasize that these accuracies are lower bounds on the actual robust accuracy, and the objective of this analysis is not to find the optimal solution to the problem of multiple perturbation adversarial training, but to elucidate the trade-off between attacking the two pipeline stages.

D MODEL ARCHITECTURE

Second-level $M_{\mathcal{A}}$ models. A key advantage of PROTECTOR is that we can build upon existing defenses against individual perturbation type. Specifically, for MNIST, we use the same CNN architecture as Zhang et al. [2019] for our $M_{\mathcal{A}}$ models, and we train these models using their proposed TRADES loss. For CIFAR-10, we use the same training setup and model architecture as Carmon et al. [2019], which is based on a robust self-training algorithm that utilizes unlabeled data to improve the model robustness.

Perturbation classifier C_{adv}. For both MNIST and CIFAR-10 datasets, the architecture of the perturbation classifier C_{adv} is similar to the individual $M_{\mathcal{A}}$ models. Specifically, for MNIST, we use the CNN architecture in Zhang et al. [2019] with four convolutional layers, followed by two fully-connected layers. For CIFAR-10, C_{adv} is a WideResNet [Zagoruyko and Komodakis, 2016] model with depth 16 and widening factor of 2 (WRN-16-2). The architectures for classifying ℓ_p perturbations and common corruptions are largely the same, except that the final classification layers have different dimensions due to the different label set sizes.

E TRAINING DETAILS

E.1 SPECIALIZED ROBUST PREDICTORS $M_{\mathcal{A}}$

MNIST. We use the Adam optimizer [Kingma and Ba, 2015] to train our models along with a piece-wise linearly varying learning rate schedule [Smith, 2018] to train our models with maximum learning rate of 10^{-3}. The base models $M_{\ell_1}, M_{\ell_2}, M_{\ell_\infty}$ are trained using the TRADES algorithm for 20 iterations, and step sizes α_1 = 2.0, α_2 = 0.3, and α_∞ = 0.05 for the $\ell_1, \ell_2, \ell_\infty$ attack types within perturbation radii ϵ_1 = 10.0, ϵ_2 = 2.0, and ϵ_∞ = 0.3 respectively.[1]

CIFAR10. The individual $M_{\mathcal{A}}$ models are trained to be robust against $\{\ell_\infty, \ell_1, \ell_2\}$ perturbations of $\{\epsilon_\infty, \epsilon_1, \epsilon_2\} = \{0.003, 10.0, 0.05\}$ respectively. For CIFAR10, the attack step sizes $\{\alpha_\infty, \alpha_1, \alpha_2\} = \{0.005, 2.0, 0.1\}$ respectively. The training of the individual $M_{\mathcal{A}}$ models is directly based on the work of Carmon et al. [2019].

E.2 PERTURBATION CLASSIFIER C_{adv}

MNIST. We train the model for 5 epochs using the SGD optimizer with weight decay as 5×10^{-4}. We used a variation of the learning rate schedule from Smith [2018], which is piecewise linear from 5×10^{-4} to 10^{-3} over the first 2 epochs, and down to 0 till the end. The batch size is set to 100 for all experiments.

CIFAR10. We train the model for 5 epochs using the SGD optimizer with weight decay as 5×10^{-4}. We used a variation of the learning rate schedule from Smith [2018], which is piecewise linear from 5×10^{-3} to 10^{-2} over the first 2 epochs, and down to 0 till the end. The batch size is set to 100 for all experiments.

Creating the Adversarial Perturbation Dataset. We create a static dataset of adversarially perturbed images and their corresponding attack label for training the perturbation classifier C_{adv}. For generating adversarial images, we perform weak adversarial attacks that are faster to compute. In particular, we perform 10 iterations of the PGD attack. For MNIST, the attack step sizes $\{\alpha_\infty, \alpha_1, \alpha_2\} = \{0.05, 2.0, 0.3\}$ respectively. For CIFAR10, the attack step sizes $\{\alpha_\infty, \alpha_1, \alpha_2\} = \{0.005, 2.0, 0.1\}$ respectively. Note that we perform the Sparse-ℓ_1 or the top-k PGD attack for the ℓ_1 perturbation ball, as introduced by Tramèr and Boneh [2019]. We set the value of k to 10, that is we move by a step size $\frac{\alpha_1}{k}$ in each of the top 10 directions with respect to the magnitude of the gradient.

CIFAR10-C. We use a dropout value of 0.3 along with the same optimizer (SGD). We use a learning rate of 0.01 and SGD optimizer for 5 epochs, with linear rate decay to 0.001 between the second epoch and the fifth epoch For experiments on classifying corruptions of severity 1, we find that the model takes longer to train. Hence, we train the model for 10 epochs, whereas all other models (at other severity levels) were trained for 5 epochs.

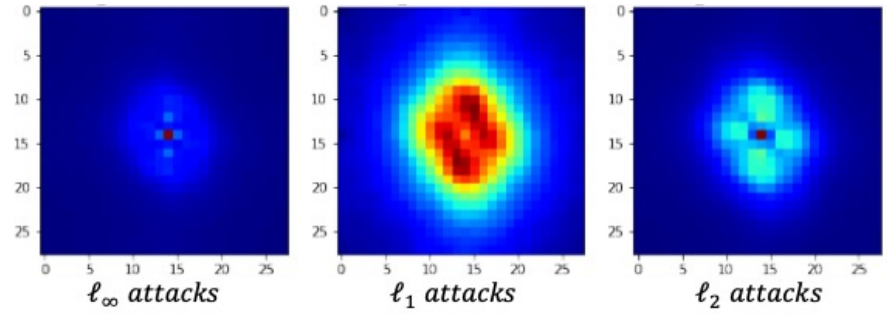
F ATTACKS USED FOR EVALUATION

A description of all the attacks used for evaluation of the models is presented here. From the AutoAttack library [Croce and Hein, 2020b], we make use of all the three variants of the Adaptive PGD attack (APGD-CE, APGD-DLR, APGD-T) along with the targeted and standard version of Fast Adaptive Boundary Attack (FAB, FAB-T) [Croce and Hein, 2020a] and the Square Attack [Andriushchenko et al., 2020]. We utilize the AA$^+$ version in the auto-attack library for stronger attacks.

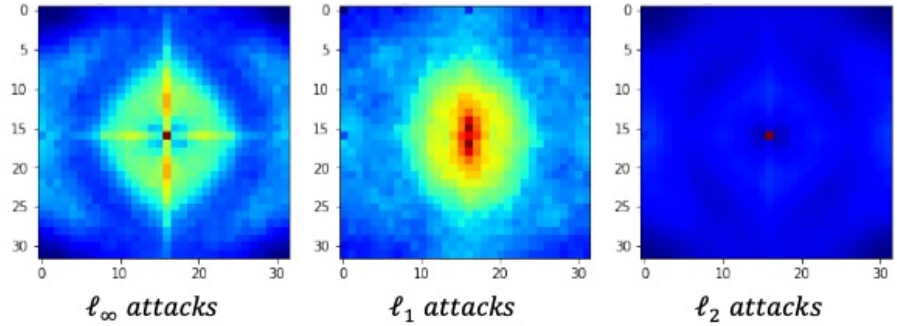
Attack Hyperparameters. For the attacks in the AutoAtack library we use the default parameter setting in the strongest available mode (such as AA$^+$). For the custom PGD attacks, we evaluate the models with 10 restarts and 200 iterations of the PGD attack. The step size of the $\{\ell_\infty, \ell_1, \ell_2\}$ PGD attacks are set as follows: For MNIST, the attack step sizes $\{\alpha_\infty, \alpha_1, \alpha_2\} = \{0.01, 1.0, 0.1\}$ respectively. For CIFAR10, the attack step sizes $\{\alpha_\infty, \alpha_1, \alpha_2\} = \{0.003, 1.0, 0.02\}$ respectively.

Further, in line with previous work [Tramèr and Boneh, 2019, Maini et al., 2020] we evaluate our models on the first 1000 images of the test set of MNIST and CIFAR-10, since many of the attacks employed are extremely computationally expensive and slow to run. Specifically, on a single GPU, the entire evaluation for a single model against all the attacks discussed with multiple restarts will take nearly 1 month, and is not feasible.

[1] We use the Sparse ℓ_1 descent [Tramèr and Boneh, 2019] for the PGD attack in the ℓ_1 constraint.



(a) MNIST dataset



(b) CIFAR10 dataset

Figure 4: We present the Fourier spectrums of various attacks on a vanilla model trained on (a) MNIST and (b) CIFAR10 datasets by averaging the per-pixel DFT over the entire test set, i.e. for an $\ell_\infty, \ell_1, \ell_2$ adversarial example corresponding to image in the test set.

G FOURIER FEATURES

Yin et al. [2019] studied various perturbations in their Fourier domain. Their work mainly focused on studying the Fourier spectrum of various common corruptions, and they showed how model robustness was affected by the data augmentation scheme used. In particular, they found that certain augmentation strategies benefit robustness to perturbations in the high frequency domain.

On the contrary, in our work, we use Fourier features to classify perturbation types. While Yin et al. [2019] directly studied only the perturbation (δ) added to the image, we visualize the Fourier transform of the actual perturbed image ($\mathbf{x} + \delta$). This makes it more challenging to distill the perturbation from the original image. Secondly, we study the Fourier transform of various adversarially crafted examples. In what follows, we will first provide a visual example to justify how adversarial examples crafted by different attack types, have different Fourier spectrums. We then utilise this property to use Fourier features as an input to the perturbation classifier for classifying the perturbation type.

Fourier Spectrum. We follow the same naming convention as Yin et al. [2019]. For an input image $\mathbf{x} \in \mathbb{R}^{d_1 \times d_2}$, we will represent the 2-dimensional discrete Fourier transform (DFT) by $\mathcal{F} : \mathbb{R}^{d_1 \times d_2} \to \mathbb{C}^{d_1 \times d_2}$. $\mathcal{F}^{-1}$ represents the inverse DFT. Since the Fourier transform belongs to the complex plane, we estimate $\mathbb{E}\left[|\mathcal{F}(\mathbf{x}_{adv})[i,j]|\right]$ by averaging over adversarial examples generate for each image in the test set.

Note that Yin et al. [2019] had estimated only the perturbation ($\mathbb{E}\left[|\mathcal{F}(\mathbf{x}_{adv} - \mathbf{x})[i,j]|\right]$) and not the perturbed image in their work. However, since at test time we do not have access to the original image, we only perform our analysis based on the perturbed input.

We present the Fourier spectrums in Figure 4. While adversarial examples typically have an imperceptible amount of perturbation for the human eye, the visualization of these adversarial examples through the Fourier spectrums help us visually distinguish between them. We also note that the Fourier spectrum for each attack does not show similar characteristics across different datasets (MNIST and CIFAR10). However, the characteristics stay consistent when independently attacking a given model on the same dataset.

Table 2: **Vanilla Model:** Empirical overlap of ℓ_{p,ϵ_p} attack perturbations in different ℓ_{q,ϵ_q} regions for (a) MNIST $(\epsilon_1, \epsilon_2, \epsilon_\infty) = (10, 2.0, 0.3)$; (b) CIFAR-10 $(\epsilon_1, \epsilon_2, \epsilon_\infty) = (10, 0.5, 0.03)$. Each column represents the range (min - max) of ℓ_q norm for perturbations generated using ℓ_p PGD attack.

Attack	**MNIST**			**CIFAR10**		
	$\ell_\infty < 0.3$	$\ell_2 < 2.0$	$\ell_1 < 10$	$\ell_\infty < 0.03$	$\ell_2 < 0.5$	$\ell_1 < 10$
PGD ℓ_∞	≤ 0.3	(3.67 - 6.05)	(54.8 - 140.9)	≤ 0.03	(1.33 - 1.59)	(62.7 - 85.5)
PGD ℓ_2	(0.40 - 0.86)	≤ 2.0	(11.2 - 24.1)	(0.037 - 0.10)	≤ 0.05	(15.4 - 20.9)
Sparse ℓ_1	(0.70 - 1.0)	(2.08 - 2.92)	≤ 10.0	(0.27 - 0.77)	(1.32 - 1.88)	≤ 10.0

Table 3: **PROTECTOR:** Empirical overlap of ℓ_{p,ϵ_p} attack perturbations in different ℓ_{q,ϵ_q} regions for (a) MNIST $(\epsilon_1, \epsilon_2, \epsilon_\infty) = (10, 2.0, 0.3)$; (b) CIFAR-10 $(\epsilon_1, \epsilon_2, \epsilon_\infty) = (10, 0.5, 0.03)$. Each column represents the range (min - max) of ℓ_q norm for perturbations generated using ℓ_p PGD attack.

Attack	**MNIST**			**CIFAR10**		
	$\ell_\infty < 0.3$	$\ell_2 < 2.0$	$\ell_1 < 10$	$\ell_\infty < 0.03$	$\ell_2 < 0.5$	$\ell_1 < 10$
PGD ℓ_∞	≤ 0.3	(5.03-6.12)	(100.40-138.52)	≤ 0.03	(1.46-1.69)	(73.15-93.26)
PGD ℓ_2	(0.35-0.95)	$\leq$2.0	(17.06-27.88)	(0.036-0.29)	$\leq$0.05	(5.83-21.21)
Sparse ℓ_1	(0.81-1.0)	(2.13-2.98)	$\leq$10.0	(0.42-1.0)	(1.50-2.91)	$\leq$10.0

We use this observation to augment PROTECTOR with an ensemble of diverse perturbation classifiers. We do so by training another model C_{adv} for which the inputs are *only* the Fourier features of the corresponding adversarial examples. The training process and architecture for such a classifier stays identical as one that classifies adversarial examples in their image domain.

H PERTURBATION CATEGORIZATION

H.1 EMPIRICAL PERTURBATION OVERLAP

While we justify the choice of perturbation sizes in our theoretical proofs in Appendix B.4 and C.2, in this section we demonstrate the empirical agreement of the choices of perturbation sizes we make for our results on MNIST and CIFAR10 datasets. To measure how often adversarial perturbations of different attacks overlap, we empirically quantify the overlapping regions by attacking a benign model with PGD attacks. In Table 2 we report the range of the norm of perturbations in the alternate perturbation region for any given attack type. The observed overlap is exactly 0% in all cases and the observation is consistent across MNIST and CIFAR10 datasets.

Table 4: Perturbation type classification accuracy for different perturbation types. The perturbation classifier C_{adv} is trained on adversarial examples against two $M_{\mathcal{A}}$ models. Each column represent the model used to create transfer-based attack via the attack type in the corresponding row. The represented accuracy is an aggregate over 1000 randomly sampled attacks of the $\ell_\infty, \ell_2, \ell_1$ types for the corresponding algorithms (and datasets).

	M_{ℓ_∞}	M_{ℓ_2}	M_{ℓ_1}	MAX	AVG	MSD
MNIST-PGD	100%	100%	99.3%	99.0%	99.6%	99.1%
MNIST-AutoAttack	100%	100%	99.0%	99.5%	100%	100%
CIFAR10-PGD	99.9%	99.5%	100%	100%	98.7%	95.7%
CIFAR10-AutoAttack	99.9%	99.9%	100%	100%	99.7%	99.7%

To contrast the results with that of attacking a vanilla model, we also present results on the perturbation overlap when we attack PROTECTOR with PGD attacks (in Table 3). It is noteworthy that the presence of a perturbation classifier forces the adversaries to generate such attacks that increase the norm of the perturbations in alternate ℓ_q region. Secondly, we also observe that in the case of CIFAR10, the ℓ_2 PGD attack has a large overlap with the ℓ_1 norm of radius 10. However, recall that in case of ℓ_2 attacks for CIFAR10, both the base models M_{ℓ_1} and M_{ℓ_∞} were satisfactorily robust. Hence, the attacker has no incentive to reduce the perturbation radius for an ℓ_q norm since the perturbation classifier only performs a binary classification between ℓ_1 and ℓ_∞ attacks.

H.2 ROBUSTNESS OF C_{adv}

In this section, we present the results of the perturbation type classifier C_{adv} against transfer adversaries. The results for the robustness of the perturbation classifier C_{adv} in the presence of adaptive adversaries is presented in Table 4. Note that C_{adv} transfers well across the board, even if the adversarial examples are generated against new models that are unseen for C_{adv} during training, achieving extremely high test accuracy. Further, even if the adversarial attack was generated by a different algorithm such as from the AutoAttack library, the transfer success of C_{adv} still holds up. In particular, the obtained accuracy is $> 95\%$ across all the individual test sets created. The attack classification accuracy is in general highest against those generated by attacking M_{ℓ_1} or M_{ℓ_∞} for CIFAR10, and M_{ℓ_2} or M_{ℓ_∞} for MNIST. This is an expected consequence of the nature of generation of the static dataset for training the perturbation classifier C_{adv} as described in Section 5.1.

Table 5: Classification accuracy for common corruptions at different severity levels. The task is a 19 class classification problem. In the training setting "Combined", all images of different severity levels are used for training. The model predicts the corruption type among the 19 possible corruptions.

	Tested on				
Training	Level 1	Level 2	Level 3	Level 4	Level 5
Level Specific	87.2%	97.7%	97.0%	98.7%	99.5%
Combined	85.4%	96.2%	97.2%	98.1%	99.1%

H.3 MORE RESULTS ON COMMON CORRUPTIONS

For each image in the original CIFAR-10 test set, CIFAR-10-C includes corrupted images of 19 different corruption types at 5 severity levels. In this section, we present results on corruption classification at different severity levels. Specifically, we train a single model on images of all severity levels. Then to evaluate on each of the 5 severity levels, we also train another model on corrupted images of the same level. As mentioned in Section 6.1, each corruption type has 9K training samples at each severity level, and 1K for testing. We ensure that all corrupted samples of the same original CIFAR-10 image are in the same data split, so that no sample in the test split corresponds to the same original image in the training split.

We present the corruption type classification accuracies at different severity levels in Table 5. We observe that the classification accuracy is around 90% for all severity levels, even when the severity level is low and the corruptions are hard to notice for the human eye. Note that for a 19-class classification problem, random guessing would only yield about 5% accuracy. Further, the test accuracy increases as the severity of the corruption increases. This can be explained due the fact that increasing the magnitude of corruptions makes them more representative and easier to be distinguished from others. Note that models trained on standard image classification tasks are typically more resilient to corruptions at a lower severity, and images with a high corruption severity can be detrimental to the prediction performance of standard classifiers. Therefore, it is important to correctly identify such highly corrupted images. We also note that a combined model trained on multiple corruption severity levels does not have a significant trade-off in test accuracy to those trained on the specific levels. Specifically, the drop in test set accuracy varies between 0.4% and 1.8% across various severity levels, and the decrease is much less noticeable when the severity level becomes large.

I ADAPTIVE ATTACKS

I.1 AGGREGATING PREDICTIONS FROM DIFFERENT $M_{\mathcal{A}}$ AT INFERENCE

In all our experiments in this work the adversary constructs adversarial examples using the softmax based adaptive strategy for aggregating predictions from different $M_{\mathcal{A}}$ models, as described in Equation 4 for the column 'Ours' and using the 'max' strategy (Equation 3) for results described in the column 'Ours*'.

However, for consistency of our defense strategy irrespective of the attacker's strategy, the defender only utilizes predictions from the specialized model $M_{\mathcal{A}}$ corresponding to the most-likely attack (Equation 3) to provide the final prediction (only forward propagation) for generated adversarial examples. In our evaluation, we found a negligible impact of changing this aggregation to the 'softmax' strategy for aggregating the predictions. For example, we show representative results in case of the APGD (ℓ_∞, ℓ_2) attacks on the CIFAR10 dataset in Table 6.

Table 6: Comparison between using a 'softmax' based aggregation of predictions from different specialized models versus using the prediction from the model corresponding to the most likely attack (only at inference time). Results are presented for APGD ℓ_2, ℓ_∞ attacks on the CIFAR10 dataset.

Attack	Max-approach (Eq. 3)	Softmax-approach (Eq. 4)
APGD-CE ℓ_2 ($\epsilon_2 = 0.5$)	75.7%	75.6%
APGD-DLR ℓ_2 ($\epsilon_2 = 0.5$)	76.5%	76.7%
APGD-CE ℓ_∞ ($\epsilon_\infty = 0.03$)	86.9%	86.9%
APGD-DLR ℓ_∞ ($\epsilon_\infty = 0.03$)	91.8%	91.2%

Table 7: Performance of Adaptive attacks that attempt to separately fool the perturbation classifier and the alternate specialized robust model. The corresponding objective functions for each attack are specified in Appendix I.

Attack	Dual Attack (Eq. 47)	Binary Attack (Eq. 48)
PGD ℓ_∞ ($\epsilon_\infty = 0.03$)	69.3%	73.2%
PGD ℓ_2 ($\epsilon_2 = 0.5$)	72.1%	74.8%
Sparse PGD ℓ_1 ($\epsilon_1 = 10$)	64.7%	59.1%

I.2 TRADE-OFF BETWEEN FOOLING $M_\mathcal{A}$ AND C_{adv}

The adversary chooses the strongest attack over a set of adaptive attacks targeted at each $M_\mathcal{A}$. For any data point (x,y) each targeted attack optimises the following constraint:

$$\begin{aligned} &\min_{\delta_p} \ell_p(x + \delta_p) \\ \text{s.t.} \quad &M_\mathcal{A}(x + \delta_p) \neq y; \quad C_{adv}(x + \delta_p) = p \end{aligned} \tag{46}$$

We perform the attack for each of the PGD attacks for $p \in \{1, 2, \infty\}$. To design the exact objective function for optimization of Equation 46, we take inspiration from a similar exploration by Carlini and Wagner [2017].

First, we combine a dual loss function for individually fooling the $M_\mathcal{A}$ model and the perturbation classifier C_{adv} by giving different importance to each of them using a parameter λ. More specifically, for an input (x, y), the objective for finding an adversarial example of type $\mathcal{A} \in \mathcal{S}$ can be written as:

$$\mathcal{L}_{(x,y,\mathcal{A})} = -1 \cdot \text{CrossEntropyLoss}(C_{adv}(x), \mathcal{A}) + \lambda \cdot \text{CrossEntropyLoss}(\mathcal{M}_\mathcal{B}(x), y) \tag{47}$$

where $\mathcal{B} = \arg\max C_{adv}(x)$. We experiment with values of $\lambda \in \{10^{-1}, 1, 10, 100\}$ and report the worst adversarial example in each case.

Secondly, we design an alternate approach where the adversary is constrained to fool the perturbation classifier (owing to a strong binary misclassification loss). It then attempts to fool the alternate $M_\mathcal{A}$ model under this constraint. More specifically, if $\mathcal{B} = \arg\max C_{adv}(x)$, then

$$\mathcal{L}_{(x,y,\mathcal{A})} = -1 \cdot (\mathcal{A} = \mathcal{B}) + \lambda \cdot \text{CrossEntropyLoss}(\mathcal{M}_\mathcal{B}(x), y) \tag{48}$$

We perform the above optimization for the PGD attacks in the $\ell_\infty, \ell_1, \ell_2$ perturbation radius constraints. In case of the ℓ_1 attack, we optimize using the stronger Sparse-ℓ_1 attack [Tramèr and Boneh, 2019]. The adversarial robustness of PROTECTOR (on CIFAR10) to these attacks is reported in Table 7. We note that the formulation used in the main paper (Equation 4) that uses a 'softmax' bridge between the two levels of the pipeline performs better than the attacks outlined above. In particular, we observe that adversaries find it difficult to balance the two losses separately in order to satisfy the dual constraint.

Table 8: Attack-wise breakdown of adversarial robustness on the MNIST dataset. *Ours* represents the PROTECTOR method against the adaptive attack strategy described in Section 5.4, and *Ours** represents the standard attack setting.

	M_{ℓ_∞}	M_{ℓ_2}	M_{ℓ_1}	MAX	AVG	MSD	Ours	Ours*
Benign Accuracy	99.2%	98.7%	98.8%	98.6%	99.1%	98.3%	98.9%	98.9%
PGD-ℓ_∞	92.8%	6.2%	0.0%	50.0%	64.8%	65.7%	83.5%	89.1%
APGD-CE	91.5%	3.6%	0.0%	41.0%	59.1%	65.2%	84.3%	84.6%
APGD-DLR	91.8%	8.0%	0.0%	43.9%	61.9%	66.0%	88.6%	88.4%
APGD-T	91.9%	2.9%	0.0%	39.6%	59.0%	64.4%	88.0%	88.6%
FAB-T	92.5%	5.0%	0.0%	48.8%	64.3%	65.5%	99.0%	98.6%
SQUARE	90.3%	7.6%	0.0%	45.9%	65.1%	68.2%	93.0%	93.3%
ℓ_∞ attacks ($\epsilon = 0.3$)	90.2%	2.6%	0.0%	39.0%	57.8%	63.5%	78.1%	79.0%
PGD-ℓ_2	84.9%	74.9%	51.6%	63.6%	69.5%	71.7%	73.0%	75.5%
DDN	42.3%	76.0%	53.1%	62.2%	64.6%	70.1%	87.5%	94.3%
APGD-CE	78.9%	74.0%	50.7%	61.9%	65.0%	69.6%	72.2%	76.4%
APGD-DLR	79.3%	75.2%	54.1%	63.2%	65.1%	70.9%	74.4%	78.2%
APGD-T	80.7%	73.8%	48.0%	61.0%	63.9%	69.6%	70.8%	74.3%
FAB-T	12.2%	74.8%	49.4%	62.5%	63.7%	69.1%	86.9%	96.3%
SQUARE	25.6%	82.3%	66.6%	71.7%	71.8%	75.0%	96.9%	96.6%
ℓ_2 attacks ($\epsilon = 2.0$)	9.5%	72.3%	47.8%	58.5%	58.6%	65.7%	66.6%	72.3%
PGD-ℓ_1	72.5%	74.6%	78.5%	52.9%	59.3%	67.9%	73.8%	79.4%
FAB-T	20.0%	71.6%	77.6%	43.9%	51.2%	67.5%	74.3%	85.0%
ℓ_1 attacks ($\epsilon = 10$)	18.8%	70.6%	77.5%	41.8%	46.1%	64.3%	68.1%	72.5%
All Attacks	7.3%	2.6%	0.0%	29.1%	37.1%	57.2%	63.6%	67.2%
Average All Attacks	69.8%	47.4%	35.3%	54.1%	63.2%	68.4%	83.1%	86.6%

J BREAKDOWN OF COMPLETE EVALUATION

Now we present a breakdown of results of the adversarial robustness of baseline approaches and PROTECTOR against all the attacks in our suite. We also report the worst case performance against the union of all attacks.

J.1 MNIST

In Table 8, we provide a breakdown of the adversarial accuracy of all the baselines, individual $M_{\mathcal{A}}$ models and the PROTECTOR method, with both the adaptive and standard attack variants on the MNIST dataset. PROTECTOR outperforms prior baselines by 6.4% on the MNIST dataset. It is important to note that PROTECTOR shows significant improvements against most attacks in the suite. Compared to the previous state-of-the-art defense against multiple perturbation types (MSD), if we compare the performance gain on each individual attack algorithm, the average accuracy increase of 14.7% on MNIST dataset. These results demonstrate that PROTECTOR considerably mitigates the trade-off in accuracy against individual attack types.

J.2 CIFAR-10

In Table 9, we provide a breakdown of the adversarial accuracy of all the baselines, individual $M_{\mathcal{A}}$ models and the PROTECTOR method, with both the adaptive and standard attack variants on the CIFAR10 dataset. PROTECTOR outperforms prior baselines by 10%. Once again, note that PROTECTOR shows significant improvements against most attacks in the suite. Compared to the previous state-of-the-art defense against multiple perturbation types (MSD), if we compare the performance gain on each individual attack algorithm, the improvement is significant, with an average accuracy increase of 14.2% on. These results demonstrate that PROTECTOR considerably mitigates the trade-off in accuracy against individual attack types. Further, PROTECTOR also retains a higher accuracy on benign images, as opposed to past defenses that have to sacrifice the benign accuracy for the robustness on multiple perturbation types. The clean accuracy of PROTECTOR is over 7% higher than such existing defenses on CIFAR-10, and the accuracy is close to $M_{\mathcal{A}}$ models trained for a single perturbation type.

Table 9: Attack-wise breakdown of adversarial robustness on CIFAR-10. *Ours* represents PROTECTOR against the adaptive attack strategy described in Section 5.4, and *Ours** represents the standard attack setting.

	M_{ℓ_∞}	M_{ℓ_2}	M_{ℓ_1}	MAX	AVG	MSD	Ours	Ours*
Benign Accuracy	89.5%	93.9%	89.0%	81.0%	84.6%	81.7%	89.0%	89.0%
PGD-ℓ_∞	62.3%	36.2%	36.0%	43.2%	41.1%	46.6%	62.3%	62.3%
APGD-CE	62.1%	35.5%	35.9%	38.5%	41.1%	46.3%	62.2%	63.9%
APGD-DLR	60.9%	38.0%	37.7%	39.1%	43.3%	46.6%	59.1%	63.8%
APGD-T	59.4%	34.9%	35.0%	36.5%	39.7%	43.8%	58.7%	62.3%
FAB-T	59.9%	35.9%	35.4%	40.8%	40.2%	44.0%	79.1%	84.7%
SQUARE	67.2%	57.7%	50.5%	51.8%	50.8%	52.1%	85.6%	80.3%
ℓ_∞ attacks ($\epsilon = 0.003$)	59.3%	34.8%	35.0%	34.9%	39.7%	43.7%	56.1%	58.4%
PGD-ℓ_2	66.5%	77.5%	72.4%	64.4%	67.7%	66.2%	69.4%	69.6%
DDN	66.9%	77.5%	72.6%	64.5%	67.7%	66.2%	83.1%	85.2%
APGD-CE	66.3%	77.4%	72.3%	64.4%	67.2%	66.1%	71.1%	70.8%
APGD-DLR	65.6%	77.6%	72.0%	63.0%	66.0%	65.3%	70.5%	70.6%
APGD-T	65.1%	77.3%	71.5%	62.1%	65.5%	64.5%	69.4%	69.6%
FAB-T	65.0%	77.4%	71.7%	62.7%	65.7%	64.5%	88.7%	90.4%
SQUARE	81.2%	86.2%	81.7%	72.0%	77.1%	72.2%	90.2%	92.1%
ℓ_2 attacks ($\epsilon = 0.5$)	64.6%	77.2%	71.5%	61.8%	65.5%	64.5%	69.3%	69.4%
PGD-ℓ_1	30.2%	48.5%	62.5%	50.8%	61.0%	58.2%	59.8%	64.1%
FAB-T	35.0%	47.2%	61.3%	48.3%	63.8%	57.7%	65.5%	69.3%
ℓ_1 attacks ($\epsilon = 10$)	27.6%	45.3%	60.9%	43.7%	60.0%	56.1%	57.9%	59.5%
All Attacks	27.6%	32.9%	35.0%	31.5%	39.3%	43.5%	53.5%	54.9%
Average All Attacks	60.9%	59.0%	57.9%	53.5%	57.2%	57.4%	71.6%	73.3%

Table 10: Effect of the number (n) of specialized robust predictors $M_{\mathcal{A}}$ in PROTECTOR(n) on CIFAR-10. The analysis was performed for an architecture that only utilizes the raw input, and not the Fourier features.

	PROTECTOR(2)	PROTECTOR(3)
Clean accuracy	90.8%	92.2%
APGD ℓ_∞ ($\epsilon = 0.03$)	64.8%	56.3%
APGD ℓ_2 ($\epsilon = 0.5$)	68.8%	69.2%
Sparse ℓ_1 ($\epsilon = 10$)	55.9%	52.3%

J.3 DIFFERENT NUMBER OF SECOND-LEVEL $M_{\mathcal{A}}$ PREDICTORS

We also evaluate PROTECTOR with three second-level predictors, i.e., M_{ℓ_1}, M_{ℓ_2} and M_{ℓ_∞}. The results are presented in Table 10. This alternative design reduces the overall accuracy of the pipeline model. We hypothesize that this happens because the M_{ℓ_1} model is already reasonably robust against the ℓ_2 attacks, as shown in Table 2b. However, having both M_{ℓ_1} and M_{ℓ_2} models allows adaptive adversaries to find larger regions for fooling both C_{adv} and $M_{\mathcal{A}}$, thus hurting the overall performance against adaptive adversaries.

References

Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *European Conference on Computer Vision*, 2020.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017.

Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled data improves adversarial robustness. In *Advances in Neural Information Processing Systems*, pages 11190–11201, 2019.

Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *International Conference on Machine Learning*, 2020a.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020b.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pages 125–136, 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Pratyush Maini, Eric Wong, and J. Zico Kolter. Adversarial robustness against the union of multiple perturbation models. In *International Conference on Machine Learning*, 2020.

Leslie N Smith. A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. *arXiv preprint arXiv:1803.09820*, 2018.

Florian Tramèr and Dan Boneh. Adversarial training and robustness for multiple perturbations. In *Advances in Neural Information Processing Systems*, pages 5866–5876, 2019.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2018.

Dong Yin, Raphael Gontijo Lopes, Jon Shlens, Ekin Dogus Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision. In *Advances in Neural Information Processing Systems*, pages 13276–13286, 2019.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pages 7472–7482, 2019.