Classifying Perturbation Types for Robustness Against Multiple Adversarial Perturbations

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

Abstract

Despite the recent advances in defenses against adversarial attacks, deep neural networks typically stay vulnerable to adversaries outside the perturbation type they are trained to be robust against. Recent work has aimed to improve the robustness of a single model against the union of multiple perturbation types, e.g., ℓ_1, ℓ_2 and ℓ_∞. However, when evaluating their accuracy against each individual perturbation type, they still do not match the performance of models trained specifically for that single perturbation type. To close this gap, we propose *Classify Then Predict (CTP)*, a two-stage pipeline to improve the robustness against the union of multiple perturbation types. Instead of training a single label predictor for different perturbation types, CTP first classifies the perturbation type of the input, and then leverages a label predictor specifically trained against that adversary to provide the final prediction. We first provide a theoretical analysis to show that adversarial examples with different perturbation types constitute different distributions, which makes it possible to distinguish them. Further, we show that at test time, the adversary faces a natural trade-off between fooling the attack classifier and the robust label predictor, and as a result, is unable to plant strong attacks against the pipeline. On MNIST, our approach achieves a 10% improvement on the overall adversarial accuracy against the union of $\ell_1, \ell_2, \ell_\infty$ perturbation balls.

1. Introduction

There has been a long line of work studying the vulnerabilities of machine learning models to small changes in the input data. In particular, most existing work focuses on the perturbations within an ℓ_p ball of a small radius surrounding the original data points (Szegedy et al., 2013; Goodfellow et al., 2015). While the majority of the prior work has aimed at achieving robustness against a single perturbation type (Madry et al., 2018; Kurakin et al., 2017; Tramèr et al., 2018; Dong et al., 2018; Zhang et al., 2019; Carmon et al., 2019), real-world deployment of machine learning models requires them to be robust against various imperceptible changes in the input, irrespective of the attack type. This is necessitated, because the adversary can always attack machine learning systems with adversarial inputs outside the perturbation type that the model was specifically trained to be robust against. Prior work has shown that when models are trained to be robust against one perturbation type, such robustness typically does not transfer to attacks of a different type (Schott et al., 2018; Kang et al., 2019). As a result, recent work has aimed at developing models that are robust against multiple perturbation types (Tramèr & Boneh, 2019; Maini et al., 2020). Specifically, these works consider adversaries limited by their ℓ_p distance from the original input for $p \in \{1, 2, \infty\}$. While these methods improve the overall robustness against multiple perturbation types, when evaluating the robustness against each perturbation type, the robustness of models trained on multiple perturbation types is still considerably worse than those trained on a single perturbation type. Further, these methods have been found to be very sensitive to small changes in hyperparameters.

In this work, we propose an alternative view that does not require a *single model* to be robust against a union of perturbation types. Instead, we suggest the use of a union of robust models to improve the overall robustness. In particular, we propose *Classify then Predict (CTP)*, a two-stage pipeline. Given a potentially adversarial input, CTP first utilizes a perturbation classifier to predict the perturbation type of the adversarial noise that has been added to the benign sample. Afterwards, CTP returns the prediction of the model that is the most robust against the predicted perturbation type.

We validate our approach from both theoretical and empirical aspects. First, we present a theoretical analysis to show that the distribution of new inputs generated by different attack types (within a class label) are significantly distinct,

[1]IIT Delhi [2]University of California, Berkeley [3]University of Illinois at Urbana-Champaign. Correspondence to: Pratyush Maini <pratyush.maini@gmail.com>, Xinyun Chen <xinyun.chen@berkeley.edu>.

Presented at the ICML 2020 Workshop on Uncertainty and Robustness in Deep Learning. Copyright 2020 by the author(s).

and can be separated by a simple Bayesian classifier. Further, we show the existence of a natural tension between attacking the perturbation classifier and the second-stage label predictors. As a result, even an *imperfect* attack classifier is sufficient to significantly improve the overall robustness of the model to multiple perturbation types.

Finally, we evaluate our approach on the MNIST dataset. Our proposed CTP pipeline improves over the previous state-of-the-art approaches by over 10% against the union of the ℓ_1, ℓ_2 and ℓ_∞ threat models for perturbation radius = $\{10, 2, 0.3\}$ respectively.

2. Classification with Multiple Perturbations

In this section, we formally define the problem of robust classification against multiple perturbation types, and motivate the existence of a classifier that can separate them.

2.1. Problem Setting

We consider the data to consist of inputs sampled from the union of two multi-variate Gaussian distributions $\mathcal{D}$, such that the input-label pairs (x,y) can be described as:

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

where $x = [x_0, x_1, \dots, x_d] \in \mathcal{R}^{d+1}$ and $\eta = \frac{\alpha}{\sqrt{d}}$, such that the absolute value of the mean for any dimension is equal for inputs sampled from both the positive and the negative labels. We adapt this problem setting from Ilyas et al. (2019) and motivate the modifications in Appendix A.1. 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 (independently) normally distributed with mean $y\eta$ and variance σ^2 each. For the purposes of this work, we assume that $\frac{\alpha}{\sigma} > 10$ (x_0 is strongly correlated) and $d > 100$ (remaining d features are weakly correlated, but together represent a strongly correlated feature).

2.2. Perturbation Types

We focus our discussion on adversaries constrained within a fixed ℓ_p ball of radius ϵ_p around the original input, for $p \in \mathcal{S} = \{1, 2, \infty\}$. Such adversaries are frequently studied in existing work, primarily on finding the optimal first-order adversaries for different perturbation types. We define $\Delta_{p,\epsilon}$ as the ℓ_p threat model of radius ϵ and $\Delta_{\mathcal{S}} = \bigcup_{p \in \mathcal{S}} \Delta_{p,\epsilon}$. For a model parametrized over θ, the objective of the adversary is to find the optimal perturbation δ^*, such that:

$$\delta^* = \arg \max_{\delta \in \Delta_{\mathcal{S}}} \ell(x + \delta; \theta) \tag{2}$$

2.3. Different adversarial perturbation types have distinct distributions

Consider a standard classifier M trained with the objective of correctly classifying the label of inputs x in $\mathcal{D}$. While the original distribution of the input data for each label is known to us, we aim to examine how adversaries confined within different perturbation regions attack the input. The goal of the adversary is to fool the label predictor M, by finding the optimal perturbation $\delta_p \; \forall p \in S$. On the other hand, we aim at separating the new distributions corresponding to adversarial inputs within each of the perturbation balls.

Theorem 1 (Separability of perturbation types)**.** *Given a binary Gaussian classifier (M) trained to classify inputs sampled from $\mathcal{D}$, the distribution of inputs $\mathcal{D}_p^y$ obtained by adding the optimal adversarial perturbation (to the input data of given class y) confined within ℓ_p balls of radius $\left\{\alpha, \frac{\alpha}{\sqrt{d}}\right\}$ for $p \in \{1, \infty\}$ respectively, can be accurately separated by a binary Gaussian classifier (C_{adv}) with a misclassification probability $P_e \leq 10^{-24}$.*

The proof idea is to develop an error bound on the maximum error in classification of perturbation type. We first find the new distribution of optimal adversarial inputs corresponding to each perturbation type for some classification label. Then C_{adv} aims to predict the perturbation type based on **only** viewing the adversarial input, and not the delta perturbation. We present the formal proof for Theorem 1 in Appendix A.

3. Perturbation Classification

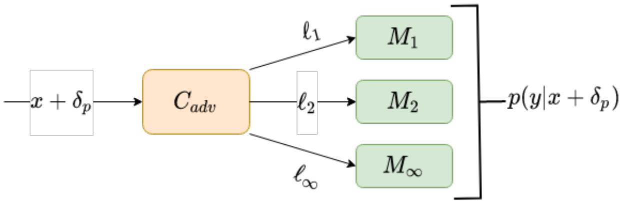

Figure 1. An overview of the Classify Then Predict (CTP) pipeline.

In Section 2, we show that the *optimal perturbations* corresponding to different perturbation types belong to distinct data distributions, and it is fairly easy to separate them using a simple classifier. However, in an adaptive white box setting, at test time the adversary has knowledge of both the perturbation classifier C_{adv} and the base models M_p. Therefore, we now evaluate if it is possible for the entire pipeline to stay robust in the presence of a dynamic adversary that attacks the entire pipeline and not the base models alone.

The classification problem consists of two tasks: **(1)** Predicting the correct class label of an adversarially perturbed (or benign) input using adversarially robust classifier M_p; and

(2) Predicting the type of adversarial perturbation that the input was subjected to using attack classifier C_{adv}.

3.1. The *Classify then Predict* Model

An illustrative explanation of the *'classify then predict'* model can be found in Figure 1. The following terminologies will be used throughout this work:

$\boldsymbol{M_p}$ M_{p,ϵ_p} refers to a model that was specifically trained to be robust against perturbations within the ℓ_p ball of radius ϵ_p via the procedure of adversarial training (Goodfellow et al., 2015). For notational convenience, we abbreviate this as M_p for future discussion. The choice of perturbation radius for the ℓ_1, ℓ_∞ regions is motivated in Appendix B.2 and set to $\boldsymbol{\epsilon_1} = \alpha + 2\sigma$ and $\boldsymbol{\epsilon_\infty} = \frac{\alpha+2\sigma}{\sqrt{d}}$. In Appendix B.3, we show that an adversarially robust model M_p, trained on $\mathcal{D}$ can achieve robust accuracy of greater than 99% against the attack type it was trained for. On the contrary, when subjected to attacks outside the trained perturbation ball, the accuracy reduces to under 2%.

$\boldsymbol{C_{adv}}$ The top level attack classifier is represented by C_{adv}. Given an input x, it outputs confidence values corresponding to each perturbation type $p \in \mathcal{S}$.

$\boldsymbol{f_\theta}$ We consider the *'classify then predict'* pipeline to be represented by f_θ. Then for any given input x,

$$\begin{aligned} c &= \operatorname{softmax}(C_{adv}(x)) \\ f_\theta(x) &= \sum_{p \in \mathcal{S}} c_p \cdot M_p(x) \end{aligned} \tag{3}$$

where c_p denotes the probability of the input belonging to perturbation type p and $M_p(x)$ is the corresponding output of the respective adversarially robust model.

Note: We make use of a combination of the results from each of the base models rather than only utilizing the confidence values from the *most-likely* model alone. This is done because the latter approach is non-differentiable across the pipeline junction and makes it impossible for gradient based adversaries to attack the perturbation classifier.

3.2. Adversarial Trade-off

All ℓ_p perturbation balls contain some overlapping region between them. Trivially, every adversary can output $\delta_p = 0$ such that there is no change in the adversarial input. It is clearly not possible for the attack classifier C_{adv} to correctly classify the attack in such a scenario. However, all the base models can correctly classify unperturbed inputs with a high probability. Therefore, it is important to examine the robustness of the entire pipeline together, against a dynamic adversary.

Theorem 2 (Adversarial trade-off)**.** *Given a data distribution $\mathcal{D}$, adversarially trained models M_{p,ϵ_p}, and an attack classifier C_{adv} that distinguishes perturbations of different ℓ_p attack types for $p \in \{1, \infty\}$, the worst case adversary can not together fool the perturbation classifier C_{adv} and the individual robust models M_p. The attack success of the adversary over the entire pipeline $\boldsymbol{f}$, $\boldsymbol{P}_e < 0.01$ for $\boldsymbol{\epsilon_1} = \alpha + 2\sigma$ and $\boldsymbol{\epsilon_\infty} = \frac{\alpha+2\sigma}{\sqrt{d}}$.*

The underlying proof idea is that if an adversary wants to fool the top level classifier and make it predict the wrong perturbation type, its attack success against the second level model M_p will suffer a negative trade-off. This suggests that in order to fool the attack classifier, any adversary must make the perturbation 'less representative' of typical 'optimal' attacks within that perturbation type. As a result, the adversary has to subject to weaker adversarial perturbations which are unable to fool the alternate M_p model as well (by possibly belonging to the overlapping region). We present the formal proof for Theorem 2 in Appendix B.

4. Empirical Evaluation

In this section, we present the results of evaluation of the pipelined *'classify then predict'* model on the MNIST dataset and contrast it with previous approaches.

4.1. Experimental Setup

Architecture. We use a 4-layer convolutional neural network for the individual M_p models on the MNIST dataset as also used by past approaches (Tramèr & Boneh, 2019; Maini et al., 2020). For the perturbation classifier C_{adv}, we once again utilize the same architecture with $|\mathcal{S}|$ output labels. Further information about the various training and attack hyperparameters can be found in Appendix C.

Attack generation. Following prior work (Schott et al., 2018; Maini et al., 2020), we utilize a comprehensive suite of attacks to evaluate the worst-case robustness of the models, which include both gradient-based and gradient-free attacks. Specifically, apart from ℓ_1, ℓ_2 and ℓ_∞ PGD adversaries, we also evaluate the following attacks from the Foolbox library (Rauber et al., 2017) for different perturbation types. (1) For ℓ_1 perturbations, we include the Salt & Pepper Attack (SAPA) (Rauber et al., 2017) and Pointwise Attack (PA) (Schott et al., 2018). (2) For ℓ_2 perturbations, we include the Gaussian noise attack (Rauber et al., 2017), Boundary Attack (Brendel et al., 2018), DeepFool (Moosavi-Dezfooli et al., 2016), Pointwise Attack (PA) (Schott et al., 2018), DDN attack (Rony et al., 2019), and Carlini&Wagner attacks (Carlini & Wagner, 2017). (3) For ℓ_∞ perturbations, we include FGSM attack (Goodfellow et al., 2015) and the Momentum Iterative Method (Dong et al., 2018).

Table 1. Summary of adversarial accuracy on the MNIST dataset(higher is better)

	M_∞	M_2	M_1	B-ABS	ABS	MAX	AVG	MSD	CTP
Clean Accuracy	99.1%	99.2%	99.3%	99%	99%	98.6%	99.1%	98.3%	99.2%
ℓ_∞ attacks ($\epsilon = 0.3$)	90.3%	0.4%	0.0%	77%	8%	51.0%	65.2%	62.7%	87.3%
ℓ_2 attacks ($\epsilon = 2.0$)	13.6%	69.2%	38.5%	39%	80%	61.9%	60.1%	67.9%	68.7%
ℓ_1 attacks ($\epsilon = 10$)	4.2%	43.4%	70.0%	82%	78%	52.6%	39.2%	65.0%	69.2%
All Attacks	3.7%	0.4%	0.0%	39%	8%	42.1%	34.9%	58.4%	**68.5**%

We perform 10 random restarts to generate stronger attacks at test time. Following prior work(Tramèr & Boneh, 2019; Maini et al., 2020), the radius of the perturbation balls for the $\{\ell_1, \ell_2, \ell_\infty\}$ perturbation regions is $\{10, 2, 0.3\}$.

4.2. Baselines

We present comparisons over related work that aims to achieve robustness against multiple perturbation types. All of these works consider the union of $\ell_1, \ell_2, \ell_\infty$ adversaries.[1]

ABS, B-ABS Schott et al. (2018) proposed the use of multiple variational autoencoders to achieve robustness to multiple perturbation types on the MNIST dataset. This method is called *analysis by synthesis* (**ABS**). Further, **B-ABS** refers to the binarized version of the same architecture.

MAX, AVG Tramèr & Boneh (2019) propose simple combinations of multiple adversarial attack types into the threat model. In the **MAX** approach, the training data is augmented with the adversarial image that leads to maximum increase in loss of the model output among all the adversarial images generated for the individual threat models. While in the **AVG** method, the training data is augmented with the adversarial images corresponding to all the attack types and then the model parameters are updated to minimize the average loss over all the perturbation types.

MSD Maini et al. (2020) showed that a simple modification to the standard PGD training procedure can result in increased robustness to multiple perturbation types. More specifically, at each iteration of the PGD attack, the steepest descent is performed in the direction that maximizes the loss among all perturbation types. This method is called **MSD**.

For each of the baselines, we report results directly from the relevant comparison drawn in Maini et al. (2020). Further, M_1, M_2, M_∞ refer to models trained using adversarial training with the PGD algorithm used to augment the data with perturbations in the $\ell_1, \ell_2, \ell_\infty$ regions respectively.

4.3. Results

We provide a summary of the worst-case performance against all attacks within a given perturbation type in Table 1. The *all attacks* metric evaluates the robustness of the models to the union of all perturbation types, that is, given any image, if we can generate an adversarial attack that leads to the wrong prediction using any approach mentioned in Section 4.1, we mark it as a failure of the model.

Our CTP pipeline improves over the prior baselines in four important ways: **(1)** CTP retains a higher accuracy against benign images, as opposed to past approaches which have to sacrifice benign accuracy in order to make a single model robust to multiple perturbation types. **(2)** CTP significantly improves on the worst case accuracy against the union of all perturbation types by a margin of over 10%. **(3)** CTP mitigates the trade-off in accuracy against individual perturbation types. For instance, against ℓ_∞ attacks our approach gains by greater than 25% accuracy as opposed to the previous state-of-art approach against the union of all attack types (MSD). **(4)** As new advances are made towards improving the robustness of models to a given perturbation type, the new model M_p can simply be plugged into the existing pipeline to replace the previous one,

5. Conclusion

In this work, we show that it is indeed possible to separate out perturbations corresponding to different attack types with high accuracy. We use this as a motivation to propose an alternative approach towards achieving robustness against the union of multiple perturbation models, which we call Classify Then Predict (CTP). We elaborate the existence of a natural tension for any adversary trying to fool the pipeline – between fooling the attack classifier and the label predictor. We use this idea to show that while it may be trivially possible for adversaries to fool the attack classifier, even an *imperfect* attack classifier is sufficient for a robust pipeline. Our results on the MNIST dataset complement our theoretical analysis, and it remains a part of future work to extend the empirical benefits of our approach to more datasets. Most importantly, the unified approach is simple to train, requires much less training time for the attack classifier and allows the research community to focus on developing better solutions to robustness against specific attack types that can be easily plugged into our framework, without the need for additional training.

[1] Schott et al. (2018) consider the ℓ_0 distance. However, it is subsumed within the ℓ_1 ball of same radius.

References

Brendel, W., Rauber, J., and Bethge, M. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations*, 2018.

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

Carmon, Y., Raghunathan, A., Schmidt, L., Duchi, J. C., and Liang, P. S. Unlabeled data improves adversarial robustness. In *Advances in Neural Information Processing Systems*, pp. 11190–11201, 2019.

Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., and Li, J. Boosting adversarial attacks with momentum. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pp. 125–136, 2019.

Kang, D., Sun, Y., Hendrycks, D., Brown, T., and Steinhardt, J. Testing robustness against unforeseen adversaries, 2019.

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

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial examples in the physical world. *ICLR Workshop*, 2017. URL https://arxiv.org/abs/1607.02533.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

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

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

Rauber, J., Brendel, W., and Bethge, M. Foolbox: A python toolbox to benchmark the robustness of machine learning models. In *Reliable Machine Learning in the Wild Workshop, 34th International Conference on Machine Learning*, 2017. URL http://arxiv.org/abs/1707.04131.

Rony, J., Hafemann, L. G., Oliveira, L. S., Ayed, I. B., Sabourin, R., and Granger, E. Decoupling direction and norm for efficient gradient-based l2 adversarial attacks and defenses. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4322–4330, 2019.

Schott, L., Rauber, J., Bethge, M., and Brendel, W. Towards the first adversarially robust neural network model on mnist. In *International Conference on Learning Representations*, 2018.

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

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks, 2013.

Tramèr, F. and Boneh, D. Adversarial training and robustness for multiple perturbations. In *Advances in Neural Information Processing Systems*, pp. 5866–5876, 2019.

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations*, 2018.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2018.

Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L., and Jordan, M. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pp. 7472–7482, 2019.

A. Separability of perturbation types (Theorem 1)

In this section, we prove the separability of adversarial inputs belonging to a given class. First, we formally define the problem setting and motivate the choices in Appendix A.1. Then, in Appendix A.2 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 A.3. Finally, in Appendix A.4 we define the distinction in the adversarial input distribution and calculate the error in classification of these adversarial input types in Appendix A.5 to conclude the proof of Theorem 1.

A.1. Problem Setting

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_p; and **(2)** Predicting the type of adversarial perturbation that the input image was subjected to using attack classifier C_{adv}.

In this section, our goal is to evaluate whether 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 the 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.

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:

$$\begin{gathered} y \overset{u.a.r}{\sim} \{-1, +1\}, \\ x_0 {\sim} \mathcal{N}(y\alpha, \sigma^2), \quad x_1, \dots, x_d \overset{i.i.d}{\sim} \mathcal{N}(y\eta, \sigma^2) \end{gathered} \tag{4}$$

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; $\boldsymbol{\mu} = [\alpha, \eta, \dots, \eta] \in \mathcal{R}^{+(d+1)}$ 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 1 each. We adapt this setting from Tsipras et al. (2018) 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).

A.2. Binary Gaussian Classifier

We assume for the purposes of this work 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:

$$p(x|y = y_i) = \frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(x - y_i.\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(x - y_i.\boldsymbol{\mu})\right), \quad \forall y_i \in \{-1, 1\} \tag{5}$$

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:

$$\begin{gathered} p(y = 1)p(x|y = 1) \overset{y=1}{>} p(y = -1)p(x|y = -1) \\ p(y = 1)p(x|y = 1) \overset{y=-1}{<} p(y = -1)p(x|y = -1) \end{gathered} \tag{6}$$

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

$$\begin{aligned}
&0 \overset{y=1}{<} x^\top A x - 2b^\top x + c \\
&A = \mathbf{\Sigma}_1^{-1} - \mathbf{\Sigma}_2^{-1} \\
&b = \mathbf{\Sigma}_1^{-1}\mu_1 - \mathbf{\Sigma}_2^{-1}\mu_2 \\
&c = \mu_1^\top \mathbf{\Sigma}_1^{-1}\mu_1 - \mu_2^\top \mathbf{\Sigma}_2^{-1}\mu_2 + \log\frac{\|\Sigma_1\|}{\|\Sigma_2\|} - 2\log\frac{p(y=1)}{p(y=-1)}
\end{aligned} \tag{7}$$

Substituting (5) and (6) in (7), we find that the optimal Bayesian decision rule for our problem is given by:

$$x^\top \boldsymbol{\mu} \overset{y=1}{>} 0 \tag{8}$$

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:

$$\begin{aligned}
\mathbf{W}_0 = \frac{\alpha}{\sqrt{2}}, \qquad \mathbf{W}_i = \frac{\alpha}{\sqrt{2d}} \quad \forall i \in \{1, \ldots, d\} \\
M^W(x) = x^\top W
\end{aligned} \tag{9}$$

A.3. 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 2 (and not an adversarially robust model). The parameters of our model are defined in (9).

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:

$$\begin{aligned}
\ell(x+\delta, y; M^W) &= -y \times M^W(x+\delta) = -yx^\top \mathbf{W} \\
\delta^* &= \arg\max_{\delta\in\Delta} \ell(x+\delta, y; M^W) \\
&= \arg\max_{\delta\in\Delta} -y(x+\delta)^\top \mathbf{W} = \arg\max_{\delta\in\Delta} -y\delta^\top \mathbf{W}
\end{aligned} \tag{10}$$

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 :

$$\delta^* = \arg\max_{\|\delta\|_p = \epsilon_p} -y\delta^\top \mathbf{W} \tag{11}$$

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

$$\begin{aligned}
\|\delta\|_p &= \left(\sum_i |\delta_i|^p\right)^{\frac{1}{p}} \\
\partial_j \|\delta\|_p &= \frac{1}{p}\left(\sum_i |\delta_i|^p\right)^{\frac{1}{p}-1} \cdot p|\delta_j|^{p-1}\operatorname{sgn}(\delta_j) = \left(\frac{|\delta_j|}{\|\delta\|_p}\right)^{p-1} \operatorname{sgn}(\delta_j)
\end{aligned} \tag{12}$$

p = 2 Making use of langrange multipliers to solve (11), we have:

$$\begin{aligned}
\nabla_\delta(-\delta^\top \Sigma^{-1}\mu) &= \lambda \nabla_\delta(\|\delta\|_p^2 - \epsilon_p^2) \\
-\mathbf{W} &= \lambda^{'} \|\delta\|_p \nabla_\delta(\|\delta\|_p)
\end{aligned} \tag{13}$$

Combining the results from (12) and replacing δ with δ_2 we obtain :

$$\begin{aligned} -\mathbf{W} &= \lambda^{'} \|\delta_2\|_2 \left(\frac{|\delta_2|}{\|\delta_2\|_2} \right) \operatorname{sgn}(\delta_2) \\ \delta_2 &= -\epsilon_2 \left(\frac{\mathbf{W}}{\|\mathbf{W}\|_2} \right) = -\epsilon_2 \mathbf{W} \end{aligned} \tag{14}$$

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

$$\begin{aligned} \delta^* &= \arg \max_{\|\delta\|_\infty = \epsilon_\infty} -y\delta^\top \mathbf{W} \\ &= \arg \max_{\|\delta\|_\infty = \epsilon_\infty} -y \sum_{i=0}^{d} \delta_i \mathbf{W}_i \end{aligned} \tag{15}$$

Since $\|\delta\|_\infty = \epsilon_\infty$, we know that $\max_i |\delta_i| = \epsilon_\infty$. Therefore (15) 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:

$$\delta_\infty = -y\epsilon_\infty \mathbf{1} \tag{16}$$

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

$$\begin{aligned} \delta^* &= \arg \max_{\|\delta\|_1 = \epsilon_1} -y \sum_{i=1}^{d} \delta_i \mathbf{W}_i \\ &= \arg \max_{\|\delta\|_1 = \epsilon_1} -y\delta_0 \mathbf{W}_0 - y \sum_{i=1}^{d} \delta_i \mathbf{W}_i \\ &= \arg \max_{\|\delta\|_1 = \epsilon_1} -y\delta_0 \frac{\alpha}{\sqrt{2}} - y \sum_{i=1}^{d} \delta_i \frac{\alpha}{\sqrt{2d}} \end{aligned} \tag{17}$$

Since $\|\delta\|_1 = \epsilon_1$, (17) is maximized when:

$$\delta_0 = -y\epsilon_1 \operatorname{sgn}(\alpha) = -y\epsilon_1, \qquad \delta_i = 0 \quad \forall i \in \{1 \ldots d\} \tag{18}$$

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).

A.4. Perturbation Classification by C_{adv}

In this section, 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:

$$\begin{aligned} X_1 &\sim \mathcal{N}(\boldsymbol{\mu} + \delta_1,\, \boldsymbol{\Sigma}) \\ X_\infty &\sim \mathcal{N}(\boldsymbol{\mu} + \delta_\infty,\, \boldsymbol{\Sigma}) \end{aligned} \tag{19}$$

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

A.5. 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 (6) the decision rule indicates the incorrect class. (This can be understood through the existence of outliers)

$$\begin{aligned} P_e &= \int P(\text{error}|x)p(x)dx \\ &= \int \min\left[p(y = \ell_1|x)p(x), p(y = \ell_\infty|x)p(x)\right] dx \end{aligned} \tag{20}$$

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 B.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 :

$$\begin{aligned} X_1 &\sim \mathcal{N}(\boldsymbol{\mu}_{adv},\, \boldsymbol{\Sigma}) \\ X_\infty &\sim \mathcal{N}(-\boldsymbol{\mu}_{adv},\, \boldsymbol{\Sigma}) \end{aligned} \tag{21}$$

We can once again combine this with the simplified Bayesian model in (8) to obtain the classification rule given by:

$$x^\top \boldsymbol{\mu}_{adv} \overset{p=1}{>} 0 \tag{22}$$

Combining the optimal perturbation definitions in (16) and (18) 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:

$$x_z = x_0 \cdot \left(1 - \frac{1}{\sqrt{d}}\right) + \frac{1}{\sqrt{d}} \sum_{i=1}^{d} x_i = \frac{2}{\alpha}\left(x^\top \boldsymbol{\mu}_{adv}\right) \tag{23}$$

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

$$\begin{aligned}
\mu_z &= \alpha\left(1-\frac{1}{\sqrt{d}}\right)+\frac{1}{\sqrt{d}}\sum_{i=1}^{d}\frac{\alpha}{\sqrt{d}} \\
&= 2\alpha-\frac{\alpha}{\sqrt{d}} \\
\sigma_z^2 &= \sigma^2\left(1+\frac{1}{d}-2\frac{1}{\sqrt{d}}+\sum_{i=1}^{d}\frac{1}{d}\right) \\
&= \sigma^2\left(2+\frac{1}{d}-2\frac{1}{\sqrt{d}}\right)
\end{aligned} \tag{24}$$

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, $z > 10$:

$$P_e \leq 10^{-24} \tag{25}$$

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 same analysis to other ℓ_p balls as well, but we consider the case of ℓ_1 and ℓ_∞ for simplicity.

B. Robustness of the *Classify then Predict* model (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_p that were specifically trained to be robust to perturbations to the input within the corresponding ℓ_p norm.

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 B.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_p. For the following discussion, it may be assumed that $\zeta_p \to 0 \; \forall p \in \{1, \infty\}$.

B.1. Binary Gaussian Classifier M_p

Extending the discussion in Appendix A.2, we now examine the learned weights of a binary Gaussian classifier M_p 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:

$$\min_{\mathbf{W}} \mathbb{E}\left[-yx^\top\mathbf{W}\right] + \frac{1}{2}\lambda||\mathbf{W}||_2^2 \tag{26}$$

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 :

$$\mathbf{W}_i^* = \mathbf{W}_j^* = \mathbf{W}_\mathbf{M} \quad \forall i, j \in \{1, \ldots, d\} \tag{27}$$

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_p 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:

$$\begin{gathered}\min_{\mathbf{W}} \max_{\|\delta\|_p \le \epsilon_p} \mathbb{E}\left[\mathbf{W}_0 \cdot (x_0 + \delta_0) + \mathbf{W_M} \cdot \sum_{i=1}^{d} (x_i + \delta_i)\right] + \frac{1}{2}\lambda\|\mathbf{W}\|_2^2 \\ \min_{\mathbf{W}} \left\{ -1 \left(\mathbf{W_0}\alpha + d \times \mathbf{W_M}\frac{\alpha}{\sqrt{d}} \right) + \frac{1}{2}\lambda\|\mathbf{W}\|_2^2 + \max_{\|\delta\|_p \le \epsilon_p} \mathbb{E}\left[-y \left(\mathbf{W}_0\delta_0 + \mathbf{W_M}\sum_{i=1}^{d} \delta_i \right) \right] \right\}\end{gathered} \tag{28}$$

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,

$$\min_{\mathbf{W_M}} \left\{ -1 \left(\alpha\sqrt{1 - d \cdot \mathbf{W_M}^2} + d \times \mathbf{W_M}\frac{\alpha}{\sqrt{d}} \right) + \max_{\|\delta\|_p \le \epsilon_p} \mathbb{E}\left[-y \left(\delta_0\sqrt{1 - d \cdot \mathbf{W_M}^2} + \mathbf{W_M}\sum_{i=1}^{d} \delta_i \right) \right] \right\} \tag{29}$$

p = ∞ As discussed in (16) the optimal perturbation δ_∞ is given by $-y\epsilon_\infty\mathbf{1}$. The optimization equation is simplified to:

$$\min_{\mathbf{W_M}} \left\{ (\epsilon_\infty - \alpha)\sqrt{1 - d \cdot \mathbf{W_M}^2} + d \times \mathbf{W_M} \left(\epsilon_\infty - \frac{\alpha}{\sqrt{d}} \right) \right\} \tag{30}$$

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:

$$\lim_{\zeta_\infty \to 0} \mathbf{W_M} = 0 \tag{31}$$

Therefore, the classifier weights are given by $\mathbf{W} = [\mathbf{W}_0, \mathbf{W}_1, \dots, \mathbf{W}_d] = [1, 0, \dots, 0]$. We also show later in Appendix B.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 (18) 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

$$\min_{\mathbf{W}} \left\{ \mathbf{W_0}(\epsilon_1 - \alpha) - d \times \mathbf{W_M}\frac{\alpha}{\sqrt{d}} + \frac{1}{2}\lambda\|\mathbf{W}\|_2^2 \right\}, \tag{32}$$

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

$$\min_{\mathbf{W}} \left\{ -\mathbf{W_0}\alpha - \mathbf{W_M}\left(\sqrt{d}\alpha - \epsilon_1\right) + \frac{1}{2}\lambda\|\mathbf{W}\|_2^2 \right\} \tag{33}$$

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_1 by considering the optimal solution when $zeta_1 = 0$. The optimal solution is then given by:

$$\lim_{\zeta_1 \to 0} \mathbf{W_M} = 1 \tag{34}$$

For the robust classifier M_1, the weights $\mathbf{W} = [\mathbf{W}_0, \mathbf{W}_1, \dots, \mathbf{W}_d] = [0, \frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}, \dots, \frac{1}{\sqrt{d}}]$. 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 B.3 that the model achieves greater than 99% accuracy against ℓ_1 adversaries for the chosen values of ζ_1.

B.2. Perturbation Size

Now that we exactly know the weights of the learned robust classifiers M_1 and M_∞, 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_∞ 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_1 model is $\sim 98\%$.

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

$$\begin{aligned}
P_{\infty,1} &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot M_\infty(x+\delta_1) > 0] \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x+\delta_1)^\top \mathbf{W} > 0] \\
&\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}[x_0 > \epsilon_1] \\
z &= \frac{\epsilon_1 - \alpha}{\sigma} = \frac{\alpha + \zeta_1 - \alpha}{\sigma} = \frac{\zeta_1}{\sigma} = 2 \\
\zeta_1 &= 2\sigma \\
\epsilon_1 &= \alpha + 2\sigma
\end{aligned} \tag{35}$$

To simplify the discussion for the M_1 model, we define a meta-feature x_M as:

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

which is distributed as :

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

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

$$\begin{aligned}
P_{1,\infty} &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot M_1(x+\delta_\infty) > 0] \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x+\delta_\infty)^\top \mathbf{W} > 0] \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot \frac{1}{\sqrt{d}} \sum_{i=1}^{d} (x_i + \delta_\infty(i)) > 0] \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x_M - \sqrt{d} \cdot \epsilon_\infty) > 0] \\
&\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[x_M > \sqrt{d} \cdot \epsilon_\infty\right] \\
z &= \frac{\sqrt{d} \cdot \epsilon_\infty - \alpha}{\sigma} = \frac{\alpha + \sqrt{d} \cdot \zeta_\infty - \alpha}{\sigma} = \frac{\sqrt{d} \cdot \zeta_\infty}{\sigma} = 2 \\
\zeta_\infty &= \frac{2\sigma}{\sqrt{d}} \\
\epsilon_\infty &= \frac{\alpha + 2\sigma}{\sqrt{d}}
\end{aligned} \tag{36}$$

B.3. Robustness of individual M_p 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.

$$d \geq 100, \qquad \frac{\alpha}{\sigma} \geq 10 \tag{37}$$

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

p = ∞

$$\begin{aligned}
P_{\infty,\infty} &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot M_\infty(x+\delta_\infty) > 0] \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x+\delta_\infty)^\top \mathbf{W} > 0] \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x_0+\delta_\infty(0)) > 0] \\
&\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}[x_0 > \epsilon_\infty] \\
z &= \frac{\epsilon_\infty - \alpha}{\sigma} = \frac{\alpha}{\sigma}\left(\frac{1}{\sqrt{d}} - 1\right) + \frac{2}{\sqrt{d}}
\end{aligned} \tag{38}$$

using the assumptions in (37),

$$P_{\infty,\infty} \geq 0.999 \tag{39}$$

p = 1 For a given input-label pair $(x+\delta_1, y)$,

$$\begin{aligned}
P_{1,1} &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot M_1(x+\delta_1) > 0] \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x+\delta_1)^\top \mathbf{W} > 0] \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot \frac{1}{\sqrt{d}}\sum_{i=1}^{d}(x_i+\delta_1(i)) > 0] \\
&= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu},\boldsymbol{\Sigma})}[y \cdot (x_M+\delta_M) > 0] \\
&\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[x_M > \frac{\epsilon_1}{\sqrt{d}}\right] \\
z &= \frac{\frac{\epsilon_1}{\sqrt{d}} - \alpha}{\sigma} = \frac{\alpha}{\sigma}\left(\frac{1}{\sqrt{d}} - 1\right) + \frac{2}{\sqrt{d}}
\end{aligned} \tag{40}$$

using the assumptions in (37),

$$P_{1,1} \geq 0.999 \tag{41}$$

B.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 following scenarios:

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

Table 2. 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. The table shows the range of the values that the mean can take depending on the decision taken by the adversary.

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.

$$C_{adv}(x) = \begin{cases} 1, & \text{if} \quad ||x_0| - \alpha| < \epsilon_\infty + \frac{\alpha}{2} \\ 0, & \text{otherwise} \end{cases} \tag{42}$$

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.

If we consider a black-box setting where the adversary has no knowledge of the classifier C_{adv}, and can only attack M_p it is easy to see that the proposed pipeline obtains a high adversarial accuracy against the union of ℓ_1 and ℓ_∞ perturbations:

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.

B.5. Trade-off between attacking M_p 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:

$$\begin{aligned}
\mathbb{P}\left[f(x)=y\right] &= \mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right]\mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[y\cdot M_\infty(x)>0|C_{adv}(x)\right]\\
&\qquad + (1-\mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right])\mathbb{P}_{x\sim\mathcal{N}(y\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[y\cdot M_1(x)>0|\neg C_{adv}(x)\right]\\
&= \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right]\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_\infty(x)>0|C_{adv}(x)\right]\\
&\qquad + (1-\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right])\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_1(x)>0|\neg C_{adv}(x)\right]
\end{aligned} \tag{43}$$

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_∞ 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.

$$\begin{aligned}
\mathbb{P}\left[f(x)=y\right] &= \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right]\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_\infty(x)>0|C_{adv}(x)\right]\\
&\qquad + (1-\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right])\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_1(x)>0|\neg C_{adv}(x)\right]\\
&\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right]\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_\infty(x)>0|C_{adv}(x)\right]\\
&\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[|x_0-\alpha|\leq\frac{\alpha}{2}-\epsilon_\infty\right]\\
&\geq 2\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[x_0\leq\alpha-\frac{\alpha}{2}+\epsilon_\infty\right]\\
z = \frac{(\alpha-\frac{\alpha}{2}+\epsilon_\infty)-\alpha}{\sigma} &= -\frac{\alpha}{2\sigma}+\frac{3\sigma}{2\sigma\sqrt{d}}
\end{aligned} \tag{44}$$

using the assumptions in (37),

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

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_∞ model. But we know from (39) that the M_∞ 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_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.

$$\begin{aligned}\mathbb{P}\left[f(x)=y\right] &= \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right]\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_\infty(x)>0|C_{adv}(x)\right]\\ &\quad+(1-\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[C_{adv}(x)\right])\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu}_{adv},\boldsymbol{\Sigma})}\left[M_1(x)>0|\neg C_{adv}(x)\right]\\ &\geq \mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[-\alpha-\frac{\alpha}{2}-\epsilon_1\leq x_0\leq-\alpha+\frac{\alpha}{2}+\epsilon_1\right]\\ &\quad+0.999(\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[x_0<-\alpha-\frac{\alpha}{2}-\epsilon_1\text{ or }x_0>-\alpha+\frac{\alpha}{2}+\epsilon_1\right])\\ &\geq 0.999(\mathbb{P}_{x\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})}\left[x_0<-\alpha-\frac{\alpha}{2}-\epsilon_1\text{ or }x_0>-\alpha+\frac{\alpha}{2}+\epsilon_1\right])\end{aligned} \tag{46}$$

using the assumptions in (37),

$$\mathbb{P}\left[f(x)=y\right]\sim 0.99 \tag{47}$$

This concludes the proof for Theorem 2, showing that an adversary can not stage successful attacks on the entire pipeline and faces a natural tension between attacking the label predictor and the attack classifier. Finally, we emphasize that the shown 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 expose the existence of the trade-off between attacking the two stages of the pipeline.

C. Training Hyperparameters

We use the Adam optimizer (Kingma & 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_1, M_2, M_∞ are trained using the PGD algorithm with 10 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.[2]

At test time, the number of iterations of the PGD algorithm are increased to 100 for each of the adversaries and we perform 10 random restarts to attain the most successful adversarial perturbation.

[2] We use the Sparse ℓ_1 descent (Tramèr & Boneh, 2019) for the PGD attack in the ℓ_1 constraint.