Stratified Adversarial Robustness with Rejection

Jiefeng Chen [*1] **Jayaram Raghuram** [*1] **Jihye Choi** [1] **Xi Wu** [2] **Yingyu Liang** [1] **Somesh Jha** [1]

Abstract

Recently, there is an emerging interest in adversarially training a classifier with a rejection option (also known as a selective classifier) for boosting adversarial robustness. While rejection can incur a cost in many applications, existing studies typically associate zero cost with rejecting perturbed inputs, which can result in the rejection of numerous slightly-perturbed inputs that could be correctly classified. In this work, we study adversarially-robust classification with rejection in the stratified rejection setting, where the rejection cost is modeled by rejection loss functions monotonically non-increasing in the perturbation magnitude. We theoretically analyze the stratified rejection setting and propose a novel defense method – *Adversarial Training with Consistent Prediction-based Rejection* (CPR) – for building a robust selective classifier. Experiments on image datasets demonstrate that the proposed method significantly outperforms existing methods under strong adaptive attacks. For instance, on CIFAR-10, CPR reduces the total robust loss (for different rejection losses) by at least 7.3% under both seen and unseen attacks.

1. Introduction

Building robust models against adversarial attacks is critical for designing secure and reliable machine learning systems (Biggio et al., 2013; Szegedy et al., 2014; Biggio & Roli, 2018; Madry et al., 2018; Zhang et al., 2019). However, the robust error of existing methods on complex datasets is still not satisfactory (*e.g.*, (Croce et al., 2020)). Also, the robust models usually have poor generalization to threat models that are not utilized during training (Stutz et al., 2020; Laidlaw et al., 2021). Given these limitations, it is important to design selective classifiers that know when to reject or abstain from predicting on adversarial examples. This can be especially crucial when it comes to real-world, safety-critical systems such as self-driving cars, where abstaining from prediction is often a much safer alternative than making an incorrect decision. Along this line of adversarial robustness with rejection, several recent studies (Laidlaw & Feizi, 2019; Stutz et al., 2020; Sheikholeslami et al., 2021; Pang et al., 2022; Tramèr, 2022; Kato et al., 2020) have extended the standard definition of robust error to the setting where the classifier can also reject inputs, and they consider rejecting any perturbed input to be a valid decision that does not count towards the robust error.

A key limitation with these studies is that they associate *zero cost* with the rejection decision on perturbed inputs [1], whereas rejection can often have a high cost in many practical applications. For example, consider a selective classifier for traffic signs in a self-driving system. If it rejects an input (*e.g.*, a perturbed "speed limit 60" sign), then the system may not know how to react and thus need human intervention (*e.g.*, adjust the speed). In such cases, rejection has the cost of service-denial and manual intervention (Markoff, 2016; Cunningham & Regan, 2015; Mozannar & Sontag, 2020). In contrast to the practical consideration, existing studies on adversarial robustness with rejection typically do not explicitly consider a cost for rejecting perturbed inputs. The learned models thus may not satisfy the need of these applications. Indeed, the models from existing methods may end up rejecting too many slightly-perturbed inputs that could be correctly classified. As a concrete example, on MNIST with ℓ_∞-perturbation magnitude 0.4, the method CCAT (Stutz et al., 2020) achieves very good performance on the existing metrics such as 1.82% rejection rate on clean test inputs and 75.50% robust accuracy with detection (a metric introduced in (Tramèr, 2022); see Eq. 2). However, for 99.30% of the test points, CCAT will reject some small perturbations within magnitude as small as 0.02. More results can be found in our experiments in Section 5. In summary, while rejecting such small perturbations has a cost, existing studies have not adequately measured the quality of the selective classifiers, and the training methods may not learn desired models for such applications.

To address this limitation, we revisit adversarially-robust

[*]Equal contribution [1]Department of Computer Sciences, University of Wisconsin at Madison [2]Google. Correspondence to: Jiefeng Chen <jiefeng@cs.wisc.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

[1]Note that the rejection of clean inputs does incur a cost.

classification with rejection by introducing rejection loss functions to model the potential cost of rejection. This offers a more flexible framework than the traditional adversarial robustness studies that do not consider rejection (roughly equivalent to associating with rejection the same loss as mis-classification), and recent studies on adversarial robustness with rejection (e.g. (Tramèr, 2022)) that associate zero loss with rejecting any perturbed input. We focus on *the stratified rejection setting where the rejection loss functions are monotonically non-increasing in the perturbation magnitude*. This is motivated by the consideration that small input perturbations should not be rejected when correct classification is possible, and thus their rejection should incur a large cost. However, large input perturbations can often be harder to classify, and rejection may be the best option when correct classification is not possible; so they should incur a lower rejection cost compared to smaller perturbations. Furthermore, we consider the challenging scenario where *the rejection loss function used for testing is unknown at training time*. That is, the learning method is not designed for one specific rejection loss; the learned selective classifier should work under a range of reasonable rejection loss functions. Our goal is then to design such a method that can learn a robust selective classifier with small loss due to *both* mis-classification and rejection.

In summary, we make the following contributions:

- We propose to introduce rejection loss functions to model the potential cost of rejection in applications, and study the stratified rejection setting with monotonically non-increasing rejection loss functions (Section 2).
- We provide a theoretical analysis of the stratified rejection setting. We analyze the existence of a robust selective classifier and discuss conditions when it can improve over classifiers without rejection (Section 3).
- We propose a novel defense method CPR inspired by our theoretical analysis. Our experiments demonstrate that CPR significantly outperforms previous methods under strong adaptive attacks (Sections 4 and 5). Its performance is strong for different rejection losses, on both traditional and our new metrics, and under seen and unseen attacks. CPR can be combined with different kinds of adversarial training methods (e.g., TRADES (Zhang et al., 2019)) to enhance their robustness.

1.1. Related Work

Adversarial robustness of deep learning models has received significant attention in recent years. Many defenses have been proposed and most of them have been broken by strong adaptive attacks (Athalye et al., 2018; Tramèr et al., 2020). The most effective approach for improving adversarial robustness is adversarial training (Madry et al., 2018; Zhang et al., 2019). However, adversarial training still does not achieve very high robust accuracy on complex datasets. For example, as reported in RobustBench (Croce et al., 2020), even state-of-the-art adversarially trained models struggle to exceed 67% robust test accuracy on CIFAR-10.

One approach to break this adversarial robustness bottleneck is to allow the classifier to reject inputs, instead of trying to correctly classify all of them. Standard (non-adversarial) classification with a reject option (or selective classification) has been extensively studied in the literature (Tax & Duin, 2008; Geifman & El-Yaniv, 2019; Charoenphakdee et al., 2021; Cortes et al., 2016). Selective classification in the transductive setting with provable guarantees has been studied by Goldwasser et al. (2020). Recently, there has been a great interest in exploring adversarially robust classification with a reject option (Laidlaw & Feizi, 2019; Stutz et al., 2020; Kato et al., 2020; Yin et al., 2020; Sheikholeslami et al., 2021; Tramèr, 2022; Pang et al., 2022; Balcan et al., 2023). We next discuss some of these closely related works .

Stutz et al. (2020) proposed to adversarially train confidence-calibrated models using label smoothing and confidence thresholding so that they can generalize to unseen adversarial attacks. Sheikholeslami et al. (2021) modified existing certified-defense mechanisms to allow the classifier to either robustly classify or detect adversarial attacks, and showed that it can lead to better certified robustness, especially for large perturbation sizes. Pang et al. (2022) observed that two coupled metrics, the prediction confidence and the true confidence (T-Con), can be combined to provably distinguish correctly-classified inputs from mis-classified inputs. Based on this, they propose to learn a rectified confidence (R-Con) that models T-Con, which is then used to adversarially train a selective classifier. Laidlaw & Feizi (2019) proposed a method called Combined Abstention Robustness Learning (CARL) for jointly learning a classifier and the region of the input space on which it should abstain, and showed that training with CARL can result in a more accurate and robust classifier. In (Balcan et al., 2023), the authors introduced a random feature subspace threat model and showed that classifiers without the ability to abstain (reject) are provably vulnerable to this adversary; but allowing the classifier to abstain (e.g., via a thresholded nearest-neighbor algorithm) can overcome such attacks.

An important aspect that has not been explored in prior works is the cost or loss of rejecting perturbed inputs. This is important for designing robust classifiers that do not reject many slightly-perturbed inputs which could be correctly classified. To the best of our knowledge, we are the first to study adversarially-robust classification with rejection in the stratified-rejection setting.

2. Stratified Adversarial Robustness with Rejection

Notations. Let $\mathcal{X} \subseteq \mathbb{R}^d$ denote the space of inputs $\mathbf{x}$ and $\overline{\mathcal{Y}} := \{1, \cdots, k\}$ denote the space of outputs y. Let $\mathcal{Y} := \overline{\mathcal{Y}} \cup \{\perp\}$ be the extended output space where $\perp$ denotes the abstain or rejection option. Let Δ_k denote the set of output probabilities over $\overline{\mathcal{Y}}$ (*i.e.*, the simplex in k-dimensions). Let $d(\mathbf{x}, \mathbf{x}')$ be a norm-induced distance on $\mathcal{X}$ (*e.g.*, the ℓ_p-distance for some $p \geq 1$), and let $\mathcal{N}(\mathbf{x}, r) := \{\mathbf{x}' \in \mathcal{X} : d(\mathbf{x}', \mathbf{x}) \leq r\}$ denote the neighborhood of $\mathbf{x}$ with distance r. Let $\wedge$ and $\vee$ denote the min and max operations respectively (reducing to AND and OR operations when applied on boolean values). Let $\mathbf{1}\{c\}$ define the binary indicator function which takes value 1 (0) when the condition c is true (false).

Review. We first review the standard setting of adversarial robustness without rejection and the setting with rejection in a recent line of work. Given samples from a distribution $\mathcal{D}$ over $\mathcal{X} \times \overline{\mathcal{Y}}$, the goal is to learn a classifier with a reject option (a selective classifier), $f : \mathcal{X} \to \mathcal{Y}$, that has a small error. The *standard robust error* at adversarial budget $\epsilon > 0$ is defined as (Madry et al., 2018):

$$R_\epsilon(f) := \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \left[\max_{\mathbf{x}' \in \mathcal{N}(\mathbf{x},\epsilon)} \mathbf{1}\{f(\mathbf{x}') \neq y\} \right], \quad (1)$$

which does not allow rejection (*i.e.*, rejection is an error). A few recent studies (e.g. (Tramèr, 2022)) have proposed the *robust error with detection* at adversarial budget ϵ as

$$R_\epsilon^{\text{rej}}(f) := \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \Big[\mathbf{1}\{f(\mathbf{x}) \neq y\} \vee \max_{\mathbf{x}' \in \mathcal{N}(\mathbf{x},\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') \notin \{y, \perp\}\big\} \Big], \quad (2)$$

which allows the rejection of small (even infinitesimal) perturbations without incurring any error.

Rejection Loss Functions. The above studies are not well-suited for the needs of certain applications where rejection can have a cost. We would like to associate with the reject decision a loss that is a function of the perturbation magnitude. Intuitively, rejecting a clean input should incur the maximum loss, and the loss of rejecting a perturbed input should decrease (or at least not increase) as the perturbation magnitude increases. Formally, let $\ell^{\text{rej}}(r) : [0, \infty) \to [0, 1]$ be a function specifying the loss of rejecting a perturbation $\mathbf{x}'$ of a clean input $\mathbf{x}$ with perturbation magnitude $r = d(\mathbf{x}, \mathbf{x}')$. We consider two concrete cases of such losses. The *step rejection loss* is defined as

$$\ell^{\text{rej}}(r) = \mathbf{1}\{r \leq \alpha_0 \epsilon\} \quad (3)$$

for some $\alpha_0 \in [0, 1]$. That is, rejecting a perturbed input of magnitude smaller than $\alpha_0\epsilon$ has a loss 1 but rejecting larger perturbations has no loss. The *ramp rejection loss* is defined as follows for some $t \geq 0$

$$\ell^{\text{rej}}(r) = \left(1 - \frac{r}{\epsilon}\right)^t. \quad (4)$$

For instance, for $t = 1$, the ramp rejection loss decreases linearly with the perturbation magnitude.

Total Robust Loss. With the rejection loss modeling the potential cost of rejection, the adversary can make the selective classifier suffer a mis-classification loss when there exists a perturbation $\mathbf{x}'$ with $f(\mathbf{x}') \notin \{y, \perp\}$, or suffer a rejection loss $\ell^{\text{rej}}(d(\mathbf{x}, \mathbf{x}'))$ when there exists a perturbation $\mathbf{x}'$ that is rejected (*i.e.*, $f(\mathbf{x}') = \perp$). Then our goal is to learn a selective classifier that has a small total loss due to both mis-classification and rejection induced by the adversary, which is formalized as follows.

Definition 1. The total robust loss *of a selective classifier f at adversarial budget $\epsilon > 0$ with respect to a given rejection loss function ℓ^{rej} is:*

$$L_\epsilon(f; \ell^{rej}) := \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \Big[\max_{\mathbf{x}' \in \mathcal{N}(\mathbf{x},\epsilon)} \Big(\mathbf{1}\{f(\mathbf{x}') \notin \{y, \perp\}\} \vee \mathbf{1}\{f(\mathbf{x}') = \perp\}\, \ell^{rej}(d(\mathbf{x}, \mathbf{x}')) \Big) \Big].$$

Here $\vee$ denotes the maximum of the mis-classification loss and the rejection loss. While this definition is compatible with any rejection loss, we will focus on the monotonically non-increasing ones. This definition also applies to classifiers without a rejection option, in which case the total robust loss reduces to the standard robust error $R_\epsilon(f)$.

The Curve of Robust Error. The definition of the total robust loss however depends on the specific instantiation of ℓ^{rej}, which may vary for different applications. In practice, we would like to have a single evaluation of f which can be combined with different definitions of ℓ^{rej} to compute the total robust loss. Towards this end, we propose the notion of the *curve of robust error (or accuracy)*, which generalizes the existing metrics. More importantly, Lemma 1 shows that the curve can be used to compute the total robust loss for different rejection losses (proof given in Appendix B.2).

Definition 2. The curve of robust error *of a selective classifier f at adversarial budget $\epsilon \geq 0$ is $\{R_\epsilon^{rej}(f, \alpha) : \alpha \in [0, 1]\}$, where*

$$R_\epsilon^{rej}(f, \alpha) := \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \Big[\max_{\mathbf{x}' \in \mathcal{N}(\mathbf{x},\, \alpha\epsilon)} \mathbf{1}\{f(\mathbf{x}') \neq y\} \vee \max_{\mathbf{x}'' \in \mathcal{N}(\mathbf{x},\epsilon)} \mathbf{1}\big\{f(\mathbf{x}'') \notin \{y, \perp\}\big\} \Big]. \quad (5)$$

The curve of robust accuracy *or simply* the robustness curve *of f at adversarial budget ϵ is defined as*

$$\{A_\epsilon^{rej}(f, \alpha) := 1 - R_\epsilon^{rej}(f, \alpha) : \alpha \in [0, 1]\}. \quad (6)$$

We name $R_\epsilon^{\text{rej}}(f, \alpha)$ the *robust error with rejection* at α, and $A_\epsilon^{\text{rej}}(f, \alpha)$ the *robustness with rejection* at α. The intuition behind $R_\epsilon^{\text{rej}}(f, \alpha)$ for a fixed α is clear. For small perturbations within $\mathcal{N}(\mathbf{x}, \alpha\epsilon)$, both an incorrect prediction and rejection are considered an error. For larger perturbations outside $\mathcal{N}(\mathbf{x}, \alpha\epsilon)$, rejection is *not* considered to be an error (*i.e.*, the classifier can either classify correctly or reject larger perturbations). Moreover, $R_\epsilon^{\text{rej}}(f, \alpha)$ actually includes several existing metrics as special cases:

- When $\alpha = 1$, the metric reduces to the standard robust error at budget ϵ in Eq. (1), *i.e.*, $R_\epsilon^{\text{rej}}(f, 1) = R_\epsilon(f)$.
- When $\alpha = 0$, it reduces to the robust error with detection at budget ϵ in Eq. (2), *i.e.*, $R_\epsilon^{\text{rej}}(f, 0) = R_\epsilon^{\text{rej}}(f)$. Here rejection incurs an error *only* for clean inputs.
- For any classifier f *without rejection* and any $\alpha \in [0, 1]$, it reduces to the standard robust error at budget ϵ defined in Eq. (1), *i.e.*, $R_\epsilon^{\text{rej}}(f, \alpha) = R_\epsilon(f)$.

Lemma 1. *Let $s(\alpha) := R_\epsilon^{rej}(f, \alpha)$. Suppose the rejection loss $\ell^{rej} : [0, \infty) \mapsto [0, 1]$ is monotonically non-increasing, differentiable almost everywhere, and $\ell^{rej}(0) = 1$ and $\ell^{rej}(\epsilon) = 0$. Then the total robust loss simplifies to $L_\epsilon(f; \ell^{rej}) = -\int_0^1 s(\alpha)\, \mathrm{d}\ell^{rej}(\alpha\epsilon)$.*

Given this nice connection between the total robust loss and the curve of robust error, we advocate for evaluating a selective classifier f using the curve $\{R_\epsilon^{\text{rej}}(f, \alpha) : \alpha \in [0, 1]\}$. Without knowing the concrete definition of the rejection loss ℓ^{rej}, this curve can give a holographic evaluation of the model. Even when f is trained with a specific rejection loss ℓ^{rej} in mind, the curve is helpful in providing a more complete evaluation w.r.t. *any other* definition of the rejection loss at deployment time.

3. Theoretical Analysis

Our goal is to learn a robust selective classifier, even when the training does not know the precise specification of the rejection loss for testing. Some fundamental questions arise:

Q1. *Whether and under what conditions does there exist a selective classifier with a small total robust loss?* Many applications could have a cost for certain rejections, e.g., rejecting very small perturbations is undesirable. To design algorithms for such applications, we would like to first investigate the existence of a solution with a small total robust loss.

Q2. *Can allowing rejection lead to a smaller total robust loss?* The question is essentially about the benefit of selective classification over traditional adversarial robustness (without rejection) that tries to correctly classify all perturbations, typically by adversarial training or its variants.

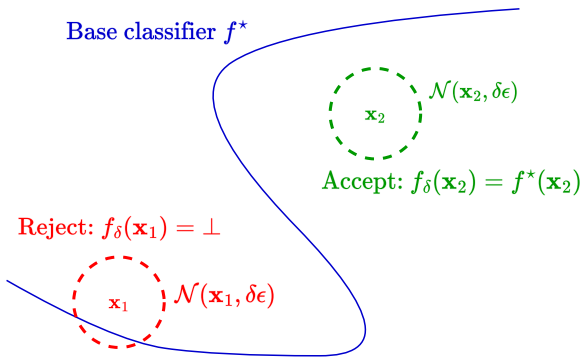

Figure 1. Proposed construction of a selective classifier f_δ from the base classifier f^* for Theorem 1. Input $\mathbf{x}_1$ is close to the boundary of f^* and is rejected, while input $\mathbf{x}_2$ is accepted.

The following theorem helps address these questions: by allowing rejection, there exists a selective classifier with a small total robust loss under proper conditions.

Theorem 1. *Consider binary classification. Let $f^*(\mathbf{x})$ be any classifier without a rejection option. For any $\delta \in [0, 1]$ and $\epsilon \geq 0$, there is a selective classifier f_δ whose robust error curve is bounded by:*

$$R_\epsilon^{rej}(f_\delta, \alpha) \leq R_{\epsilon'}(f^*), \ \forall \alpha \in [0, 1] \tag{7}$$

where $\epsilon' = \max\{(\alpha+\delta)\epsilon, (1-\delta)\epsilon\}$. Moreover, the bound is tight: for any $\alpha \in [0, 1]$, there exist simple data distributions and f^* such that any f must have $R_\epsilon^{rej}(f, \alpha) \geq R_{\epsilon'}(f^*)$.

Proof Sketch: For any $r \geq 0$, let $\mathcal{N}(f^*, r)$ denote the region within distance r to the decision boundary of f^*: $\mathcal{N}(f^*, r) := \{\mathbf{x} \in \mathcal{X} : \exists \mathbf{x}' \in \mathcal{N}(\mathbf{x}, r), f^*(\mathbf{x}') \neq f^*(\mathbf{x})\}$. Consider a parameter $\delta \in [0, 1]$ and construct f_δ as follows:

$$f_\delta(\mathbf{x}) := \begin{cases} \perp & \text{if } \mathbf{x} \in \mathcal{N}(f^*, \delta\epsilon), \\ f^*(\mathbf{x}) & \text{otherwise.} \end{cases} \tag{8}$$

This is illustrated in Fig. 1. We will show that any clean data $(\mathbf{x}, y)$ contributing error in $R_\epsilon^{\text{rej}}(f_\delta, \alpha)$ must contribute error in $R_{\epsilon'}(f^*)$, so $R_\epsilon^{\text{rej}}(f_\delta, \alpha) \leq R_{\epsilon'}(f^*)$. Intuitively, f_δ and f^* differ on the region $\mathcal{N}(f^*, \delta\epsilon)$. If f_δ gets a loss on a (possibly perturbed) input $\mathbf{x}' \in \mathcal{N}(f^*, \delta\epsilon)$, then the original input $\mathbf{x}$ is close to $\mathbf{x}'$ and thus close to the boundary of f^*. Therefore, $\mathbf{x}$ can be perturbed to cross the boundary and contribute error in $R_{\epsilon'}(f^*)$. The complete proof is given in Appendix B.3. □

Condition for Successful Selective Classification. The theorem shows that when the data allows a small robust error at budget $\epsilon' = \max\{(\alpha + \delta)\epsilon, (1 - \delta)\epsilon\}$, there is a selective classifier f_δ with a small robust error with rejection $R_\epsilon^{\text{rej}}(f_\delta, \alpha)$, which is bounded by the small robust error $R_{\epsilon'}(f^*)$. This shows a trade-off between the performance for small and large α. For $\alpha < 1 - \delta$, we have $\epsilon' < \epsilon$

and $R_\epsilon^{\text{rej}}(f_\delta, \alpha) \le R_{\epsilon'}(f^*) \le R_\epsilon(f^*)$. Note that $R_\epsilon(f^*) = R_\epsilon^{\text{rej}}(f^*, \alpha)$, so $R_\epsilon^{\text{rej}}(f_\delta, \alpha) \le R_\epsilon^{\text{rej}}(f^*, \alpha)$, i.e., f_δ can be better than f^* for such an α. However, for $\alpha \ge 1 - \delta$, the theorem does not guarantee $R_\epsilon^{\text{rej}}(f_\delta, \alpha) \le R_\epsilon(f^*)$, *i.e.*, f_δ may be worse than f^* for such an α. This trade-off is necessary in the worst case since the bound is tight, and is also observed in our experiments in Section 5. It is favorable towards a lower total robust loss when the rejection loss is monotonically non-increasing, as discussed in detail below.

Benefit of Selective Classification. The trade-off discussed above shows that by allowing rejection, the selective classifier f_δ can potentially have a smaller total robust loss than the classifier f^* without rejection. When $\alpha < 1 - \delta$, $R_\epsilon^{\text{rej}}(f_\delta, \alpha) \le R_{\epsilon'}(f^*) \le R_\epsilon(f^*) = R_\epsilon^{\text{rej}}(f^*, \alpha)$. There can be a big improvement: when a large fraction of correctly-classified clean inputs have distances in (ϵ', ϵ) to the boundary of f^*, we have $R_{\epsilon'}(f^*) \ll R_\epsilon(f^*)$ and thus $R_\epsilon^{\text{rej}}(f_\delta, \alpha) \ll R_\epsilon^{\text{rej}}(f^*, \alpha)$. This can lead to a significant improvement in the total robust loss. When $\alpha \ge 1 - \delta$, $R_\epsilon^{\text{rej}}(f, \alpha)$ may be worse than $R_\epsilon^{\text{rej}}(f^*, \alpha)$, but that only leads to milder rejection loss when ℓ^{rej} is monotonically decreasing. Then overall, the total robust loss can be improved, while the decreased amount would depend on the concrete data distribution and rejection loss.

Theorem 3 in Appendix B.4 provides a rigorous and fine-grained analysis of the robustness curve and the total robust loss of f_δ. In general, the total robust losses of f_δ and f^* only differ on points with distances larger than $(1 - \delta)\epsilon$ from the boundary of f^*. For such points, f_δ can correctly classify all their small perturbations of magnitude at most $\epsilon_0 := (1 - 2\delta)\epsilon$, and reject or correctly classify their larger perturbations. Intuitively, it correctly classifies small perturbations and rejects or correctly classifies large perturbations. Then f_δ only gets rejection loss for large magnitudes, and thus potentially gets smaller total robust losses than f^* for monotonically non-increasing rejection losses.

For a concrete example, suppose f^* has 0 standard error on the data distribution and each data point has distance at least $\frac{3\epsilon}{4}$ to the decision boundary of f^*. Then when $\delta = \frac{1}{4}$, for any data point, f_δ can correctly classify all the small perturbations with magnitude bounded by $\frac{\epsilon}{2}$, and can reject or correctly classify the larger perturbations. Then $R_\epsilon^{\text{rej}}(f_\delta, \alpha) = 0$ for $\alpha \le \frac{1}{2}$. For any step rejection loss with parameter $\alpha_0 \le \frac{1}{2}$, the total robust loss of f_δ is 0, *i.e.*, this fixed f_δ can work well for a wide range of step rejection losses. In contrast, the total robust loss of f^* is as large as $R_\epsilon(f^*)$, which is the probability mass of points within ϵ distance to the boundary of f^*. If there is a large mass of points with distance in $[\frac{3\epsilon}{4}, \epsilon]$, then f_δ has a significant advantage over f^*.

4. Proposed Defense

We propose a defense method called adversarial training with consistent prediction-based rejection (CPR), following our theoretical analysis. CPR aims to learn the f_δ in our analysis, where PGD is used to check the condition for rejection efficiently at test time. Our defense is quite general in that it can take any base classifier f^* to construct the selective classifier f_δ. Finally, to evaluate the robustness of the defense, we also discuss the design of adaptive attacks.

4.1. Consistent Prediction-Based Rejection

The CPR defense is essentially the selective classifier f_δ in Eq. (8) in Theorem 1: given a base classifier and an input $\mathbf{x}$, we define a selective classifier that rejects the input whenever the predictions in a small neighborhood of $\mathbf{x}$ are not consistent; otherwise it returns the class prediction of $\mathbf{x}$. Equivalently, it rejects inputs within a small distance to the decision boundary of the given base classifier.

To formalize the details, we introduce some notations. Consider a base classifier without rejection (i.e., f^*) realized by a network with parameters $\boldsymbol{\theta}$, whose predicted class probabilities are $\mathbf{h}(\mathbf{x}\,;\boldsymbol{\theta}) = [h_1(\mathbf{x}\,;\boldsymbol{\theta}), \cdots, h_k(\mathbf{x}\,;\boldsymbol{\theta})] \in \Delta_k$. The class prediction is $\widehat{y}(\mathbf{x}) := \operatorname{argmax}_{y \in \overline{\mathcal{Y}}} h_y(\mathbf{x}\,;\boldsymbol{\theta})$, and the cross-entropy loss is $\ell_{CE}(\mathbf{x}, y) = -\log h_y(\mathbf{x}\,;\boldsymbol{\theta})$. CPR aims to learn a selective classifier (corresponding to f_δ):

$$\mathbf{g}_{\widetilde{\epsilon}}(\mathbf{x}\,;\boldsymbol{\theta}) = \begin{cases} \mathbf{h}(\mathbf{x}\,;\boldsymbol{\theta}) & \text{if } \max\limits_{\widetilde{\mathbf{x}} \in \mathcal{N}(\mathbf{x}, \widetilde{\epsilon})} \mathbf{1}\{\widehat{y}(\widetilde{\mathbf{x}}) \ne \widehat{y}(\mathbf{x})\} = 0 \\ \perp & \text{otherwise,} \end{cases}$$

where $\widetilde{\epsilon} > 0$ is a hyper-parameter we call the *consistency radius*. To check the consistent-prediction condition efficiently, we use the projected gradient descent (PGD) method (Madry et al., 2018) to find the worst-case perturbed input $\widetilde{\mathbf{x}}$. The selective classifier is then redefined as:

$$\mathbf{g}_{\widetilde{\epsilon}}(\mathbf{x}\,;\boldsymbol{\theta}) = \begin{cases} \mathbf{h}(\mathbf{x}\,;\boldsymbol{\theta}) & \text{if } \widehat{y}(\widetilde{\mathbf{x}}) = \widehat{y}(\mathbf{x}) \\ \perp & \text{otherwise,} \end{cases} \tag{9}$$

$$\text{where } \widetilde{\mathbf{x}} = \operatorname*{argmax}_{\widetilde{\mathbf{x}} \in \mathcal{N}(\mathbf{x}, \widetilde{\epsilon})} \ell_{CE}(\widetilde{\mathbf{x}}, \widehat{y}(\mathbf{x})). \tag{10}$$

Then $\widehat{y}(\mathbf{x})$ is the predicted class when the input is accepted. The details of our CPR defense are given in Algorithm 1. Note that the PGD method in the algorithm is deterministic. Thus, the mapping $T(\mathbf{x}) = \mathbf{x}_m$ is deterministic and we can summarize the defense as: if $\widehat{y}(T(\mathbf{x})) \ne \widehat{y}(\mathbf{x})$, then reject $\mathbf{x}$; otherwise accept $\mathbf{x}$ and output the predicted class $\widehat{y}(\mathbf{x})$.

The robust base model for our defense $\mathbf{h}$ can be trained using methods such as standard adversarial training (AT) (Madry et al., 2018), TRADES (Zhang et al., 2019), and MART (Wang et al., 2020) (we will focus on the first two methods). An advantage of the proposed defense is that

Algorithm 1 CONSISTENT PREDICTION-BASED REJECTION

Input: A base model $\mathbf{h}$, a test input $\mathbf{x}$ (potentially with adversarial perturbations), the consistency radius $\widetilde{\epsilon} > 0$, the number of PGD steps $m \geq 1$, and the PGD step size $\eta > 0$.

Class prediction: $y = \widehat{y}(\mathbf{x}) = \operatorname{argmax}_{y' \in \overline{\mathcal{Y}}} h_{y'}(\mathbf{x}\,;\boldsymbol{\theta})$
Initialize the adversarial input: $\mathbf{x}_0 = \mathbf{x}$
for $i = 1, 2, \ldots, m$ **do**
  $\widehat{\mathbf{x}}_i = \mathbf{x}_{i-1} + \eta\, \mathrm{sign}\big(\nabla_{\mathbf{x}_{i-1}} \ell_{CE}(\mathbf{x}_{i-1}, y)\big)$
  $\mathbf{x}_i = \mathrm{Proj}\big(\widehat{\mathbf{x}}_i, \mathcal{N}(\mathbf{x}, \widetilde{\epsilon})\big)$ {Project $\widehat{\mathbf{x}}_i$ to $\mathcal{N}(\mathbf{x}, \widetilde{\epsilon})$}
end for
Define $T(\mathbf{x}) := \mathbf{x}_m$

Output: If $\operatorname{argmax}_{y' \in \overline{\mathcal{Y}}} h_{y'}(T(\mathbf{x})\,;\boldsymbol{\theta}) \neq y$, reject $\mathbf{x}$; otherwise, accept $\mathbf{x}$ and output the predicted class y.

it is agnostic of the rejection loss function used for evaluation, and therefore can generalize to multiple choices of the rejection loss function (see Section 5). We note that prior works such as Laidlaw & Feizi (2019) and Tramèr (2022) have also proposed the idea of using the consistency check for rejecting inputs to a classifier (equivalent to rejecting inputs based on their distance to the decision boundary). However, we are the first to systematically implement and evaluate such a defense, and propose strong adaptive attacks for evaluating it.

4.2. Adaptive Attacks

In this section, we design principled adaptive attacks to evaluate the proposed defense. By Definition 2 and Proposition 1, in order to compute the robustness curve, we need to design both inner and outer attacks and then combine them to get the final attack. We design multiple inner and outer attacks sketched below and ensemble them to get the strongest final attack. More details can be found in Appendix E

Inner Attack. For a given $\alpha \in [0, 1]$ on the robustness curve, the goal of the inner attack is to find an input $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)$ that is rejected. For CPR, this translates to finding $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)$ such that the base model has different predictions on $\mathbf{x}'$ and $T(\mathbf{x}')$.

Our first attack is Low-Confidence Inner Attack (**LCIA**), which finds $\mathbf{x}'$ by minimizing the confidence of the base model within $\mathcal{N}(\mathbf{x}, \alpha\epsilon)$. Recall that the mapping $T(\mathbf{x}')$ attempts to minimize the base model's probability on the predicted class $\widehat{y}(\mathbf{x}')$. So, if the base model has low confidence on $\mathbf{x}'$, then it will very likely have even lower probability for $\widehat{y}(\mathbf{x}')$ on $T(\mathbf{x}')$, and thus have different predictions on $T(\mathbf{x}')$ and $\mathbf{x}'$. The attack objective in this case is $\mathbf{x}' = \operatorname{argmax}_{\mathbf{z} \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)} -\log h_{\max}(\mathbf{z}\,;\boldsymbol{\theta})$, where $h_{\max}(\mathbf{z}\,;\boldsymbol{\theta}) = \max_{y \in \overline{\mathcal{Y}}} h_y(\mathbf{z}\,;\boldsymbol{\theta})$ is the prediction confidence. We use the temperature-scaled log-sum-exponential approximation to the $\max$ function in order to make it differentiable. We also consider a variant of LCIA, named Consistent-Low-Confidence Inner Attack (**CLCIA**), that minimizes the confidence of the base model on both $\mathbf{x}'$ and $T(\mathbf{x}')$ (details in Appendix E).

The third attack is Prediction-Disagreement Inner Attack (**PDIA**), an adaptive multi-target attack based on the BPDA method (Athalye et al., 2018). We use BPDA since $T(\mathbf{x})$ does not have a closed-form expression and is not differentiable. This attack considers all possible target classes and attempts to find $\mathbf{x}'$ such that the base model has high probability for the target class at $\mathbf{x}'$ and a low probability for the target class at $T(\mathbf{x}')$ (thereby encouraging rejection). The attack objective is: for each target class $j = 1, \ldots, k$,

$$\mathbf{x}'_j = \operatorname*{argmax}_{\mathbf{z} \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)} \Big[\log h_j(\mathbf{z}\,;\boldsymbol{\theta}) \,-\, \log h_j(T(\mathbf{z})\,;\boldsymbol{\theta})\Big].$$

Then we select the strongest adversarial example $\mathbf{x}'$ from the multiple target classes as follows:

$$\begin{aligned} \mathbf{x}' &= \mathbf{x}'_{j^\star} \quad \text{where} \\ j^\star &= \operatorname*{argmax}_{j \in [k]} \Big[\log h_j(\mathbf{x}'_j\,;\boldsymbol{\theta}) \,-\, \log h_j(T(\mathbf{x}'_j)\,;\boldsymbol{\theta})\Big]. \end{aligned} \tag{11}$$

Outer Attack. Given a clean input $(\mathbf{x}, y)$, the goal of the outer attack is to find $\mathbf{x}'' \in \mathcal{N}(\mathbf{x}, \epsilon)$ such that the base model has a consistent incorrect prediction with high confidence on both $\mathbf{x}''$ and $T(\mathbf{x}'')$ (ensuring that $\mathbf{x}''$ is accepted and misclassified). We propose the Consistent High-Confidence Misclassification Outer Attack (**CHCMOA**), an adaptive multi-target attack based on BPDA. The attack objective is: for each target class $j \in [k] \setminus \{y\}$,

$$\mathbf{x}''_j = \operatorname*{argmax}_{\mathbf{z} \in \mathcal{N}(\mathbf{x}, \epsilon)} \Big[\log h_j(\mathbf{z}\,;\boldsymbol{\theta}) \,+\, \log h_j(T(\mathbf{z})\,;\boldsymbol{\theta})\Big].$$

Then we select the strongest adversarial example $\mathbf{x}''$ via:

$$\begin{aligned} \mathbf{x}'' &= \mathbf{x}''_{j^\star} \quad \text{where} \\ j^\star &= \operatorname*{argmax}_{j \in [k] \setminus \{y\}} \Big[\log h_j(\mathbf{x}''_j\,;\boldsymbol{\theta}) \,+\, \log h_j(T(\mathbf{x}''_j)\,;\boldsymbol{\theta})\Big]. \end{aligned} \tag{12}$$

We also consider the High-Confidence Misclassification Outer Attack (**HCMOA**), which solves for an adversarial input $\mathbf{x}''$ such that the base model has incorrect prediction with high confidence on $\mathbf{x}''$ (details in Appendix E).

Final Attack. To get the strongest evaluation, our final attack is an *ensemble of all attacks*, including those designed above and some existing ones. We apply each attack in the ensemble with different hyper-parameters on each clean test input. If any of the attacks achieves the attack goal on an input, then the attack is considered to be successful on it.

5. Experiments

In this section, we perform experiments to evaluate the proposed method CPR and compare it with competitive baseline

methods. The code for our work can be found at `https://github.com/jfc43/stratified-adv-rej`. Our main findings are summarized as follows: **1)** CPR outperforms the baselines significantly in terms of the total robust loss with respect to different rejection losses, under both seen attacks and unseen attacks; **2)** CPR usually has significantly higher robustness with rejection compared to the baselines for small to moderate α under both seen attacks and unseen attacks; **3)** CPR also has strong performance under the traditional metrics such as robust accuracy with detection ($1 - R_{\epsilon}^{\text{rej}}(f, 0)$).

5.1. Experimental Setup

We briefly describe the experimental setup here. The detailed setup can be found in Appendix D.

Datasets. We use the MNIST (LeCun, 1998), SVHN (Netzer et al., 2011), and CIFAR-10 (Krizhevsky et al., 2009) datasets. Each dataset has a test set containing 10,000 images. Following (Stutz et al., 2020), we compute the accuracy of the models on the first 9,000 images of the test set and compute the robustness of the models on the first 1,000 images of the test set. We use the last 1,000 images of the test set as a held-out validation set for selecting the hyper-parameters of the methods (*e.g.*, the rejection threshold).

Baselines. We consider the following baselines: (1) AT + CR: adversarial training (AT) (Madry et al., 2018) with Confidence-based Rejection; (2) TRADES + CR: TRADES (Zhang et al., 2019) with Confidence-based Rejection; (3) CCAT (Stutz et al., 2020); (4) RCD (Sheikholeslami et al., 2021); (5) ATRR (Pang et al., 2022).

CPR Setup. CPR requires a base model. We consider robust base models trained using the well-known standard adversarial training (AT) (Madry et al., 2018) and TRADES (Zhang et al., 2019). Our experimental results show that CPR can boost the robustness of these models.

CPR has three hyper-parameters: the consistency radius $\tilde{\epsilon}$, the number of PGD steps m, and the PGD step size η. The $\widetilde{\epsilon}$ value controls the rejection rate of the selective classifier. We choose it such that CPR does not reject more than a small fraction of the correctly-classified clean inputs (however, it can reject a majority of the mis-classified inputs that are likely to be close to the decision boundary of the base model; rejecting such inputs is reasonable). The rejection rate of the selective classifier $\mathbf{g}_{\widetilde{\epsilon}}$ on correctly-classified clean inputs from a distribution $\mathcal{D}$ is given by $\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\mathbf{1}\{\mathbf{g}_{\widetilde{\epsilon}}(\mathbf{x}\,;\boldsymbol{\theta}) = \perp\} \mid \widehat{y}(\mathbf{x}) = y\right]$, which can be estimated using a labeled validation dataset. We choose a large enough $\widetilde{\epsilon} > 0$ such that this rejection rate is approximately p_{rej}, where p_{rej} is a user-specified rejection rate. The number of PGD steps m also affects the robustness. We use a validation set to select suitable $\tilde{\epsilon}$ and m (details in Appendix F.6). We do not tune the PGD step size η and set it to a fixed value.

On MNIST, we set $\tilde{\epsilon} = 0.1$ (such that $p_{rej} = 1\%$), $m = 20$, and $\eta = 0.01$. On SVHN and CIFAR-10, we set $\tilde{\epsilon} = 0.0055$ (such that $p_{rej} = 5\%$), $m = 10$, and $\eta = 0.001$.

Evaluation Metrics. We use the *robustness curve* at adversarial budget ϵ (Eq. (6)) and the total robust loss to evaluate all the methods. The curve is calculated for α values from the set $\{0, 0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.5, 1\}$ since in practice, we are mainly interested in the robustness with rejection for small values of α. We plot the robustness curve, with α along the x-axis and the robustness with rejection along the y-axis. We also evaluate the *total robust loss* $L_{\epsilon}(f; \ell^{\text{rej}})$ with respect to the ramp rejection loss (Eq. (4)) for $t \in \{2, 4\}$, and the step rejection loss (Eq. (3)) for $\alpha_0 \in \{0, 0.05, 0.1\}$. This gives a single metric that summarizes the robustness curve of the different methods.

Evaluation. We consider ℓ_∞-norm bounded attacks and generate adversarial examples to compute the robustness with rejection metric via *an ensemble of adversarial attacks*. The worst-case robustness is reported under the attack ensemble. A detailed discussion of the unified approach for designing strong adaptive attacks for CPR and all the baseline methods is given in Appendix E. We consider both *seen* attacks and *unseen* attacks. Suppose ϵ' is the attack perturbation budget for testing. For seen attacks, the perturbation budget $\epsilon' = \epsilon$ (on MNIST, $\epsilon' = 0.3$, while on SVHN and CIFAR-10, $\epsilon' = \frac{8}{255}$). For unseen attacks, the perturbation budget $\epsilon' > \epsilon$ (on MNIST, $\epsilon' = 0.4$, while on SVHN and CIFAR-10, $\epsilon' = \frac{16}{255}$). Finally, we use the same approach to set the rejection threshold for all the baselines. Specifically, on MNIST, we set the threshold such that only 1% of clean correctly-classified validation inputs are rejected. On SVHN and CIFAR-10, we set the threshold such that only 5% of clean correctly-classified validation inputs are rejected.

5.2. Results

Evaluating the Total Robust Loss. Table 1 compares the total robust loss of different methods for different rejection loss functions under both seen attacks and unseen attacks. The proposed method CPR outperforms the baselines significantly in almost all cases. The only exception is on MNIST for the ramp rejection loss with $t = 2$ under seen attacks, where CPR is worse than TRADES+CR and some baselines. This is because small perturbations on MNIST are easy to correctly classify and the ramp rejection loss penalizes rejecting large perturbations more, while CPR tends to reject large perturbations. In all other cases, CPR performs significantly better than the baselines. For instance, under unseen attacks, CPR reduces the total robust loss (with respect to different rejection losses) by at least 60.8%, 6.6% and 7.3% on MNIST, SVHN and CIFAR-10, respectively. Under seen attacks, CPR reduces the total robust loss by at least 18.0%

Dataset	Method	Total Robust Loss under Seen Attacks ↓					Total Robust Loss under Unseen Attacks ↓				
		Step Rej. Loss			Ramp Rej. Loss		Step Rej. Loss			Ramp Rej. Loss	
		$\alpha_0 = 0.0$	$\alpha_0 = 0.05$	$\alpha_0 = 0.1$	$t = 2$	$t = 4$	$\alpha_0 = 0.0$	$\alpha_0 = 0.05$	$\alpha_0 = 0.1$	$t = 2$	$t = 4$
MNIST	AT+CR	0.084	0.085	0.085	0.097	0.087	1.000	1.000	1.000	1.000	1.000
	TRADES+CR	0.060	0.061	0.061	**0.075**	0.065	1.000	1.000	1.000	1.000	1.000
	CCAT	0.168	0.926	1.000	0.945	0.894	0.245	0.994	1.000	0.955	0.914
	RCD	0.135	0.135	0.135	0.135	0.135	1.000	1.000	1.000	1.000	1.000
	ATRR	0.088	0.088	0.089	0.107	0.095	1.000	1.000	1.000	1.000	1.000
	AT+CPR (Ours)	**0.039**	**0.039**	**0.040**	0.133	0.059	**0.096**	**0.097**	**0.098**	**0.187**	**0.116**
	TRADES+CPR (Ours)	0.042	0.042	0.042	0.130	**0.058**	0.133	0.133	0.133	0.208	0.145
SVHN	AT+CR	0.539	0.539	0.539	0.542	0.540	0.882	0.882	0.882	0.882	0.882
	TRADES+CR	0.471	0.472	0.472	0.475	0.473	0.874	0.874	0.874	0.874	0.874
	CCAT	0.547	1.000	1.000	0.977	0.956	0.945	1.000	1.000	0.999	0.998
	RCD	0.662	0.662	0.662	0.662	0.662	0.903	0.903	0.903	0.903	0.903
	ATRR	0.552	0.552	0.552	0.566	0.556	0.885	0.885	0.885	0.891	0.887
	AT+CPR (Ours)	0.442	0.442	0.442	0.454	0.444	0.853	0.853	0.853	0.856	0.854
	TRADES+CPR (Ours)	**0.380**	**0.380**	**0.380**	**0.390**	**0.381**	**0.813**	**0.813**	**0.813**	**0.816**	**0.814**
CIFAR-10	AT+CR	0.500	0.501	0.501	0.507	0.503	0.895	0.895	0.895	0.895	0.895
	TRADES+CR	0.500	0.500	0.500	0.503	0.501	0.849	0.849	0.849	0.849	0.849
	CCAT	0.723	1.000	1.000	0.984	0.969	0.912	1.000	1.000	0.998	0.997
	RCD	0.533	0.533	0.533	0.533	0.533	0.905	0.905	0.905	0.905	0.905
	ATRR	0.512	0.513	0.513	0.518	0.515	0.887	0.887	0.887	0.887	0.887
	AT+CPR (Ours)	0.433	0.433	0.433	0.443	0.435	0.829	0.829	0.829	0.836	0.830
	TRADES+CPR (Ours)	**0.429**	**0.429**	**0.429**	**0.438**	**0.430**	**0.781**	**0.781**	**0.781**	**0.787**	**0.782**

Table 1. The total robust loss for different rejection loss functions under both seen and unseen attacks. The best result is **boldfaced**.

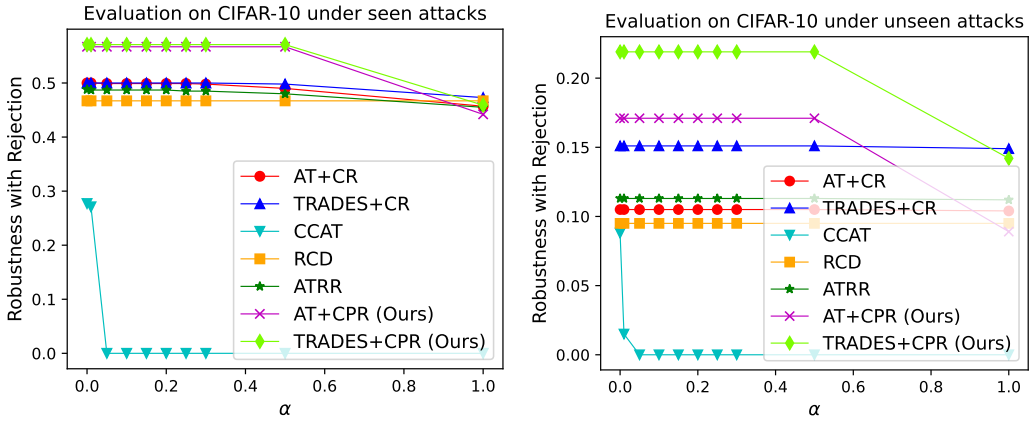

Figure 2. The robustness curves of all methods on CIFAR-10 for both seen and unseen attacks. More results are in Appendix F.2

and 12.8% on SVHN and CIFAR-10, respectively.

Evaluating the Robustness Curve. Figure 2 compares the robustness with rejection $A_\epsilon^{\text{rej}}(f, \alpha)$ of the different methods as a function of α under both seen and unseen attacks on CIFAR-10 (complete results on all the datasets can be found in Appendix F.2). Our method CPR usually has significantly higher robustness with rejection compared to the baselines for small to moderate α. The robustness with rejection of CPR only drops for large α values, which suffers less rejection loss and thus leads to smaller total robust loss (as predicted by our analysis). We also note that CCAT has worse performance than other methods since our adaptive attacks are stronger than the PGD-with-backtracking attack used in CCAT paper (Stutz et al., 2020) (see Appendix F.7).

Ablation Study on Attacks. We performed an ablation experiment to study the strength of each inner and outer attack in the attack ensemble. The results in Table 2 show that for the outer attack, CHCMOA is consistently the best across all datasets. For the inner attack, LCIA is usually the best, and CLCIA and PDIA are strong on MNIST. We emphasize that we use an ensemble of all the attacks to get the strongest final evaluation.

Ablation Study on Hyper-parameters. We performed an ablation experiment to study the effect of the hyper-parameters $\tilde{\epsilon}$ and m used by CPR. The results in Appendix F.5 show that larger $\tilde{\epsilon}$ (consistency radius) leads to better robustness with rejection at $\alpha = 0$. However, it also leads to lower robustness with rejection when α is large, which suggests that CPR rejects more perturbed inputs. Similarly, larger m (number of PGD steps) also leads

Dataset	Method	Robustness with Reject at $\alpha = 0 \downarrow$			Robustness with Reject at $\alpha = 1 \downarrow$		
		Outer Attack			Inner Attack (with Ensemble Outer Attacks)		
		AutoAttack	HCMOA	CHCMOA	LCIA	CLCIA	PDIA
MNIST	AT+CPR	97.60	97.80	**96.10**	50.90	**12.30**	65.80
	TRADES+CPR	98.10	98.40	**95.80**	7.60	5.10	**0.40**
SVHN	AT+CPR	64.20	57.50	**56.70**	**42.60**	49.40	43.90
	TRADES+CPR	71.10	63.40	**62.90**	**51.00**	56.10	54.30
CIFAR-10	AT+CPR	60.70	57.50	**57.40**	**44.40**	48.30	48.50
	TRADES+CPR	61.50	57.40	**57.20**	**45.90**	51.10	53.10

Table 2. Ablation study on the outer and inner adaptive attacks for CPR. Refer to Appendix E for more details on these attacks and the choice of metrics. All values are in percentage, and smaller values correspond to a stronger attack. **Bold** values show the strongest attacks.

to better robustness with rejection at $\alpha = 0$, but can lead to a lower robustness with rejection when α is large. We set $m = 10$ in our main experiments, which is usually sufficient for good performance, and larger values lead to minimal improvements.

Evaluating Traditional Metrics. We also evaluate different methods on the traditional metrics, including accuracy with rejection, rejection rate on clean test inputs, an F1 score-like metric (harmonic mean of accuracy-with-rejection and $1 -$ rejection rate), and the robust accuracy with detection defined in Tramèr (2022). The definitions of these metrics and the results are included in Appendix F.1. These results show that CPR has comparable performance to the baselines on clean test inputs, and also significantly outperforms the baselines w.r.t. robust accuracy with detection.

6. Conclusion and Discussion

This work studied adversarially-robust classification with rejection in the practical setting where rejection carries a loss that is monotonically non-increasing with the perturbation magnitude. We proposed the total robust loss as a generalization of the robust error for selective classifiers where rejection carries a loss, and the robustness curve as a tool to study the total robust loss. We provided an analysis of the setting and proposed a novel defense CPR for robustifying any given base model, which significantly outperforms previous methods under strong adaptive attacks.

Limitations & Future Work. In this work, we focused on rejection losses that are a function of perturbation magnitude. There could be other types of rejection losses, *e.g.*, $\ell^{\text{rej}}(r, \mathbf{x})$ that also depend on the input $\mathbf{x}$. Our defense has an increased computational cost at prediction (test) time since it requires m PGD steps to find $T(\mathbf{x})$ in Algo. 1, which requires m forward and backward passes in addition to the two forward passes required to get the predictions of $\mathbf{x}$ and $T(\mathbf{x})$. Designing defense methods that have lower computational cost and better robustness than CPR, and tackling more general rejection losses are interesting future problems.

Acknowledgements

The work is partially supported by Air Force Grant FA9550-18-1-0166, the National Science Foundation (NSF) Grants CCF-FMitF-1836978, IIS-2008559, SaTC-Frontiers-1804648, CCF-2046710 and CCF-1652140, and ARO grant number W911NF-17-1-0405. Jiefeng Chen, Jihye Choi, and Somesh Jha are partially supported by the DARPA-GARD problem under agreement number 885000. Jayaram Raghuram is partially supported through the National Science Foundation grants CNS-2112562, CNS-2107060, CNS-2003129, CNS-1838733, CNS-1647152, and the US Department of Commerce grant 70NANB21H043.

References

Athalye, A., Carlini, N., and Wagner, D. A. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pp. 274–283. PMLR, 2018. URL `http://proceedings.mlr.press/v80/athalye18a.html`.

Balcan, M.-F., Blum, A., Hanneke, S., and Sharma, D. Robustly-reliable learners under poisoning attacks. In *Conference on Learning Theory*, volume 178, pp. 4498–4534. PMLR, 2022. URL `https://proceedings.mlr.press/v178/balcan22a.html`.

Balcan, M.-F., Blum, A., Sharma, D., and Zhang, H. An analysis of robustness of non-lipschitz networks. *Journal of Machine Learning Research*, 24(98):1–43, 2023.

Biggio, B. and Roli, F. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018. doi: 10.1016/j.patcog.2018.07.023.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Srndic, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2013, Prague,*

Czech Republic, September 23-27, 2013, Proceedings, Part III, volume 8190 of *Lecture Notes in Computer Science*, pp. 387–402. Springer, 2013. doi: 10.1007/978-3-642-40994-3_25. URL https://doi.org/10.1007/978-3-642-40994-3_25.

Charoenphakdee, N., Cui, Z., Zhang, Y., and Sugiyama, M. Classification with rejection based on cost-sensitive classification. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pp. 1507–1517. PMLR, 2021. URL http://proceedings.mlr.press/v139/charoenphakdee21a.html.

Chen, J., Yoon, J., Ebrahimi, S., Arik, S. Ö., Jha, S., and Pfister, T. ASPEST: bridging the gap between active learning and selective prediction. *CoRR*, abs/2304.03870, 2023. doi: 10.48550/arXiv.2304.03870. URL https://doi.org/10.48550/arXiv.2304.03870.

Cortes, C., DeSalvo, G., and Mohri, M. Learning with rejection. In *Algorithmic Learning Theory - 27th International Conference, ALT*, volume 9925 of *Lecture Notes in Computer Science*, pp. 67–82, 2016. doi: 10.1007/978-3-319-46379-7_5. URL https://doi.org/10.1007/978-3-319-46379-7_5.

Croce, F. and Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pp. 2206–2216. PMLR, 2020.

Croce, F., Andriushchenko, M., Sehwag, V., Flammarion, N., Chiang, M., Mittal, P., and Hein, M. Robustbench: a standardized adversarial robustness benchmark. *CoRR*, abs/2010.09670, 2020. URL https://arxiv.org/abs/2010.09670.

Cunningham, M. and Regan, M. A. Autonomous vehicles: human factors issues and future research. In *Proceedings of the 2015 Australasian Road safety conference*, volume 14, 2015.

Geifman, Y. and El-Yaniv, R. SelectiveNet: A deep neural network with an integrated reject option. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2151–2159. PMLR, 2019. URL http://proceedings.mlr.press/v97/geifman19a.html.

Goldwasser, S., Kalai, A. T., Kalai, Y., and Montasser, O. Beyond perturbations: Learning guarantees with arbitrary adversarial test examples. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T. A., and Kohli, P. Scalable verified training for provably robust image classification. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pp. 4841–4850. IEEE, 2019. doi: 10.1109/ICCV.2019.00494. URL https://doi.org/10.1109/ICCV.2019.00494.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 770–778. IEEE Computer Society, 2016. doi: 10.1109/CVPR.2016.90. URL https://doi.org/10.1109/CVPR.2016.90.

Hosseini, H., Chen, Y., Kannan, S., Zhang, B., and Poovendran, R. Blocking transferability of adversarial examples in black-box learning systems. *CoRR*, abs/1703.04318, 2017. URL http://arxiv.org/abs/1703.04318.

Kato, M., Cui, Z., and Fukuhara, Y. ATRO: adversarial training with a rejection option. *CoRR*, abs/2010.12905, 2020. URL https://arxiv.org/abs/2010.12905.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Laidlaw, C. and Feizi, S. Playing it safe: Adversarial robustness with an abstain option. *CoRR*, abs/1911.11253, 2019. URL http://arxiv.org/abs/1911.11253.

Laidlaw, C., Singla, S., and Feizi, S. Perceptual adversarial robustness: Defense against unseen threat models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=dFwBosAcJkN.

LeCun, Y. The MNIST database of handwritten digits. 1998. URL http://yann.lecun.com/exdb/mnist/.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., and Jackel, L. D. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems 2, [NIPS Conference, Denver, Colorado, USA, November 27-30, 1989]*, pp. 396–404. Morgan Kaufmann, 1989.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Markoff, J. For now, self-driving cars still need humans. https://www.nytimes.com/2016/01/18/technology/driverless-cars-limits-include-human-nature.html, 2016. The New York Times.

Mozannar, H. and Sontag, D. Consistent estimators for learning to defer to an expert. In *International Conference on Machine Learning*, pp. 7076–7087. PMLR, 2020.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. 2011.

Pang, T., Yang, X., Dong, Y., Su, H., and Zhu, J. Bag of tricks for adversarial training. In *9th International Conference on Learning Representations (ICLR)*. OpenReview.net, 2021. URL https://openreview.net/forum?id=Xb8xvrtB8Ce.

Pang, T., Zhang, H., He, D., Dong, Y., Su, H., Chen, W., Zhu, J., and Liu, T.-Y. Two coupled rejection metrics can tell adversarial examples apart. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15223–15233, 2022.

Sheikholeslami, F., Lotfi, A., and Kolter, J. Z. Provably robust classification of adversarial examples with detection. In *9th International Conference on Learning Representations (ICLR)*. OpenReview.net, 2021. URL https://openreview.net/forum?id=sRA5rLNpmQc.

Stutz, D., Hein, M., and Schiele, B. Confidence-calibrated adversarial training: Generalizing to unseen attacks. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9155–9166. PMLR, 2020. URL http://proceedings.mlr.press/v119/stutz20a.html.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. Intriguing properties of neural networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6199.

Tax, D. M. J. and Duin, R. P. W. Growing a multi-class classifier with a reject option. *Pattern Recognition Letters*, 29(10):1565–1570, 2008. doi: 10.1016/j.patrec.2008.03.010. URL https://doi.org/10.1016/j.patrec.2008.03.010.

Tramèr, F. Detecting adversarial examples is (nearly) as hard as classifying them. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 21692–21702. PMLR, 2022. URL https://proceedings.mlr.press/v162/tramer22a.html.

Tramèr, F., Carlini, N., Brendel, W., and Madry, A. On adaptive attacks to adversarial example defenses. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems*, 2020.

Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., and Gu, Q. Improving adversarial robustness requires revisiting misclassified examples. In *8th International Conference on Learning Representations (ICLR)*. OpenReview.net, 2020. URL https://openreview.net/forum?id=rklOg6EFwS.

Yin, X., Kolouri, S., and Rohde, G. K. GAT: generative adversarial training for adversarial example detection and robust classification. In *8th International Conference on Learning Representations (ICLR)*. OpenReview.net, 2020. URL https://openreview.net/forum?id=SJeQEp4YDH.

Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., and Jordan, M. I. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7472–7482. PMLR, 2019. URL http://proceedings.mlr.press/v97/zhang19p.html.

Supplementary Material

In Section A, we discuss some potential societal impacts of our work. In Section B, we provide detailed proofs for the theorems and also present some additional theoretical results. In Section C, we discuss the calculation of the total robust loss for specific instantiations of the rejection loss ℓ^{rej}. In Section D, we describe the detailed setup for the experiments. In Section E, we discuss the unified approach for designing adaptive attacks for the proposed method CPR and all the baseline methods. In Section F, we present some additional experimental results.

A. Societal Impacts

Our method can build robust models with small costs due to mis-classification and rejection facing adversarial input perturbations, which can help build robust intelligent systems for applications. The proposed stratified rejection setting, which takes into account the cost of rejection, may inspire the development of improved defense methods along this line, which can have a positive impact in practice. We foresee no immediate negative societal impact.

B. Additional Theory and Proofs

B.1. Alternate Definition of $R_\epsilon^{\text{rej}}(f, \alpha)$

Proposition 1. *For any $\epsilon \geq 0$ and $\alpha \in [0, 1]$, the metric $R_\epsilon^{rej}(f, \alpha)$ can be equivalently defined as*

$$R_\epsilon^{rej}(f, \alpha) \;=\; \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \Big[\max_{\mathbf{x}'\in\mathcal{N}(\mathbf{x},\,\alpha\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') = \perp\big\} \;\vee\; \max_{\mathbf{x}''\in\mathcal{N}(\mathbf{x},\epsilon)} \mathbf{1}\big\{f(\mathbf{x}'') \notin \{y, \perp\}\big\} \Big]. \tag{13}$$

This definition allows us to interpret the metric $R_\epsilon^{rej}(f, \alpha)$ as consisting of two types of errors: 1) errors due to rejecting small perturbations within the $\alpha\epsilon$-neighborhood and 2) mis-classification errors within the ϵ-neighborhood.

Proof. Consider the original definition of $R_\epsilon^{\text{rej}}(f, \alpha)$ in Eq. (5)

$$R_\epsilon^{\text{rej}}(f, \alpha) \;=\; \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \Big[\max_{\mathbf{x}'\in\mathcal{N}(\mathbf{x},\,\alpha\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') \neq y\big\} \;\vee\; \max_{\mathbf{x}''\in\mathcal{N}(\mathbf{x},\epsilon)} \mathbf{1}\big\{f(\mathbf{x}'') \notin \{y, \perp\}\big\} \Big]. \tag{14}$$

The error in the first term inside the expectation can be split into the error due to rejection and the error due to mis-classification, *i.e.*,

$$\mathbf{1}\big\{f(\mathbf{x}') \neq y\big\} \;=\; \mathbf{1}\big\{f(\mathbf{x}') = \perp\big\} \;\vee\; \mathbf{1}\big\{f(\mathbf{x}') \notin \{y, \perp\}\big\}.$$

The maximum of this error over the $\alpha\epsilon$-neighborhood can be expressed as

$$\max_{\mathbf{x}'\in\mathcal{N}(\mathbf{x},\,\alpha\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') \neq y\big\} \;=\; \max_{\mathbf{x}'\in\mathcal{N}(\mathbf{x},\,\alpha\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') = \perp\big\} \;\vee\; \max_{\mathbf{x}'\in\mathcal{N}(\mathbf{x},\,\alpha\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') \notin \{y, \perp\}\big\},$$

which is easily verified for binary indicator functions. Substituting the above result into Eq. (14), we get

$$\begin{aligned} R_\epsilon^{\text{rej}}(f, \alpha) \;&=\; \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \Big[\max_{\mathbf{x}'\in\mathcal{N}(\mathbf{x},\,\alpha\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') = \perp\big\} \;\vee\; \max_{\mathbf{x}'\in\mathcal{N}(\mathbf{x},\,\alpha\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') \notin \{y, \perp\}\big\} \;\vee\; \max_{\mathbf{x}''\in\mathcal{N}(\mathbf{x},\epsilon)} \mathbf{1}\big\{f(\mathbf{x}'') \notin \{y, \perp\}\big\} \Big] \\ &=\; \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \Big[\max_{\mathbf{x}'\in\mathcal{N}(\mathbf{x},\,\alpha\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') = \perp\big\} \;\vee\; \max_{\mathbf{x}''\in\mathcal{N}(\mathbf{x},\epsilon)} \mathbf{1}\big\{f(\mathbf{x}'') \notin \{y, \perp\}\big\} \Big]. \end{aligned}$$

In the last step, we combined the second and third terms inside the expectation using the observation that

$$\max_{\mathbf{x}'\in\mathcal{N}(\mathbf{x},\,\alpha\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') \notin \{y, \perp\}\big\} \;\vee\; \max_{\mathbf{x}'\in\mathcal{N}(\mathbf{x},\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') \notin \{y, \perp\}\big\} \;=\; \max_{\mathbf{x}'\in\mathcal{N}(\mathbf{x},\epsilon)} \mathbf{1}\big\{f(\mathbf{x}') \notin \{y, \perp\}\big\}.$$

This shows the equivalence of the two definitions of $R_\epsilon^{\text{rej}}(f, \alpha)$. □

The definition in Eq. (13) serves as motivation for the first adaptive attack to create adversarial examples $\mathbf{x}'$ within the neighborhood $\mathcal{N}(\mathbf{x}, \alpha\epsilon)$ that are rejected by the defense method.

B.2. Proof of Lemma 1

We first prove a more general lemma in Lemma 2, from which integration by parts gives Lemma 1. Note that the step rejection loss and ramp rejection loss satisfy the conditions in Lemma 1 which gives them a simpler expression to compute the total robust loss.

Lemma 2. *Let $s(\alpha) := R_\epsilon^{rej}(f, \alpha)$ and assume it is right-semicontinuous. For any monotonically non-increasing $\ell^{rej} : [0, \infty) \to [0, 1]$, the total robust loss can be computed by:*

$$L_\epsilon(f; \ell^{rej}) \;=\; R_\epsilon^{rej}(f, 0) \;+\; \big(\ell^{rej}(0) \;-\; 1\big)\; p_{rej} \;+\; \int_0^1 \ell^{rej}(\alpha\epsilon)\, \mathrm{d}s(\alpha), \tag{15}$$

where the integral is the Riemann–Stieltjes integral, and

$$p_{rej} \;:=\; \Pr\Big[f(\mathbf{x}) = \perp \;\wedge\; \big(\forall \mathbf{x}' \in \mathcal{N}(\mathbf{x}, \epsilon),\; f(\mathbf{x}') \in \{y, \perp\}\big)\Big]$$

is the probability that the clean input $(\mathbf{x}, y)$ is rejected and no perturbations of $\mathbf{x}$ within the ϵ-ball are misclassified.

Proof. Let $\mathcal{W}$ denote the event that there exists $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \epsilon)$ such that $\mathbf{x}'$ is misclassified, i.e., $f(\mathbf{x}') \notin \{y, \perp\}$. Let $\mathcal{C}$ denote the event that the clean input $\mathbf{x}$ is rejected, i.e., $f(\mathbf{x}) = \perp$. Clearly, $p_{\text{rej}} = \Pr[\mathcal{C} \setminus \mathcal{W}]$ where $\setminus$ is the set minus. Finally, let $\mathcal{R}$ denote the event that there exists $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \epsilon)$ such that $\mathbf{x}'$ is rejected.

We only consider $\mathcal{W} \cup \mathcal{C} \cup \mathcal{R}$, since otherwise $(\mathbf{x}, y)$ contributes a loss 0 to $L_\epsilon(f; \ell^{\text{rej}})$. The union can be partitioned into three disjoint subsets.

- $\mathcal{W}$: Such an $(\mathbf{x}, y)$ contributes a loss 1. Since $R_\epsilon^{\text{rej}}(f, 0) = \Pr[\mathcal{W} \cup \mathcal{C}]$, we have $\Pr[\mathcal{W}] = R_\epsilon^{\text{rej}}(f, 0) - \Pr[\mathcal{C} \setminus \mathcal{W}] = R_\epsilon^{\text{rej}}(f, 0) - p_{\text{rej}}$. Then this subset contributes a loss $R_\epsilon^{\text{rej}}(f, 0) - p_{\text{rej}}$.
- $\mathcal{C} \setminus \mathcal{W}$: Such a data point contributes a loss $\ell^{\text{rej}}(0)$, given the assumption that ℓ^{rej} is monotonically non-increasing. Then this subset contributes a loss $\ell^{\text{rej}}(0) p_{\text{rej}}$.
- $\mathcal{R} \setminus (\mathcal{W} \cup \mathcal{C})$: That is, there exists no $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \epsilon)$ that is misclassified, the clean input $\mathbf{x}$ is accepted, but there exists some $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \epsilon)$ that is rejected. Let L_3 denote the loss contributed by this subset.

Now, it is sufficient to show that

$$L_3 = \int_0^1 \ell^{\text{rej}}(\alpha\epsilon)\, \mathrm{d}s(\alpha).$$

For any positive integer t, let $a_0 = 0 \le a_1 \le \cdots \le a_{t-1} \le a_t = 1$ be an arbitrary sequence on $[0, 1]$. Let

$$\begin{aligned}
M_i \;&=\; \sup\{\ell^{\text{rej}}(\alpha\epsilon),\;\; a_{i-1} \le \alpha \le a_i\},\\
m_i \;&=\; \inf\{\ell^{\text{rej}}(\alpha\epsilon),\;\; a_{i-1} \le \alpha \le a_i\},\\
U(s, \ell^{\text{rej}}) \;&=\; \sum_{i=1}^{t} M_i\,(s(a_i) - s(a_{i-1})),\\
L(s, \ell^{\text{rej}}) \;&=\; \sum_{i=1}^{t} m_i\,(s(a_i) - s(a_{i-1})).
\end{aligned}$$

Let $\mathcal{R}_i$ denote the event that there exists $\mathbf{x}'$ such that $a_{i-1}\epsilon < d(\mathbf{x}', \mathbf{x}) \le a_i\epsilon$ and $\mathbf{x}'$ is rejected. Then $\mathcal{R} = \cup_{i=1}^t \mathcal{R}_i$, and

$$\begin{aligned}
s(a_i) \;-\; s(a_{i-1}) \;&=\; R_\epsilon^{\text{rej}}(f, a_i) \;-\; R_\epsilon^{\text{rej}}(f, a_{i-1})\\
&=\; \Pr\big[\mathcal{R}_i \setminus (\cup_{j=1}^{i-1} \mathcal{R}_j) \setminus (\mathcal{W} \cup \mathcal{C})\big].
\end{aligned}$$

Since ℓ^{rej} is monotonically non-increasing, each data point in $\mathcal{R}_i \setminus (\cup_{j=1}^{i-1} \mathcal{R}_j) \setminus (\mathcal{W} \cup \mathcal{C})$ should contribute a loss that is within $[m_i, M_i]$. Therefore,

$$L(s, \ell^{\text{rej}}) \;\le\; L_3 \;\le\; U(s, \ell^{\text{rej}})$$

for any sequence $\{a_i\}_{i=0}^t$. When $s(\alpha)$ is right-semicontinous, the Riemann–Stieltjes integral exists, and $L_3 = \int_0^1 \ell^{\text{rej}}(\alpha\epsilon)\, \mathrm{d}s(\alpha)$. This completes the proof. □

B.3. Proof of Theorem 1

Theorem 2 (Restatement of Theorem 1)**.** *Consider binary classification. Let $f^*(\mathbf{x})$ be any classifier without a rejection option. For any $\delta \in [0,1]$ and $\epsilon \geq 0$, there exists a selective classifier f_δ, whose robust error curve is bounded by:*

$$R_\epsilon^{rej}(f_\delta, \alpha) \;\leq\; R_{\epsilon'}(f^*),\; \forall \alpha \in [0,1] \tag{16}$$

where $\epsilon' = \max\{(\alpha+\delta)\epsilon, (1-\delta)\epsilon\}$. Moreover, the bound is tight: for any $\alpha \in [0,1]$, there exist simple data distributions and f^* such that any f must have $R_\epsilon^{rej}(f,\alpha) \geq R_{\epsilon'}(f^*)$.

Proof. For any $r > 0$, let $\mathcal{N}(f^*, r)$ denote the region within distance r to the decision boundary of f^*:

$$\mathcal{N}(f^*, r) := \{\mathbf{x} \in \mathcal{X} : \exists \mathbf{x}' \in \mathcal{N}(\mathbf{x}, r) \text{ and } f^*(\mathbf{x}') \neq f^*(\mathbf{x})\}.$$

Consider a parameter $\delta \in [0,1]$ and construct a selective classifier f_δ as follows:

$$f_\delta(\mathbf{x}) \;:=\; \begin{cases} \perp & \text{if } \mathbf{x} \in \mathcal{N}(f^*, \delta\epsilon), \\ f^*(\mathbf{x}) & \text{otherwise.} \end{cases} \tag{17}$$

We will show that any sample $(\mathbf{x}, y)$ contributing error to $R_\epsilon^{\text{rej}}(f_\delta, \alpha)$ must also contribute error to $R_{\epsilon'}(f^*)$, where $\epsilon' = \max\{(\alpha+\delta)\epsilon, (1-\delta)\epsilon\}$. This will prove that $R_\epsilon^{\text{rej}}(f_\delta, \alpha) \leq R_{\epsilon'}(f^*)$.

Consider the following two cases:

- Consider the first type of error in $R_\epsilon^{\text{rej}}(f_\delta, \alpha)$: $\max_{\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)} \mathbf{1}\left[f_\delta(\mathbf{x}') \neq y\right] = 1$. This implies that there exists $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)$ such that $f_\delta(\mathbf{x}') \neq y$. So there are two subcases to consider:

  (1) $\mathbf{x}' \in \mathcal{N}(f^*, \delta\epsilon)$: in this case $\mathbf{x} \in \mathcal{N}(f^*, (\delta+\alpha)\epsilon)$.

  (2) $f^*(\mathbf{x}') \neq y$: in this case either $f^*(\mathbf{x}) \neq y$, or $f^*(\mathbf{x}) = y \neq f^*(\mathbf{x}')$ and thus $\mathbf{x} \in \mathcal{N}(f^*, \alpha\epsilon)$.

  In summary, either $f^*(\mathbf{x}) \neq y$ or $\mathbf{x} \in \mathcal{N}(f^*, (\alpha+\delta)\epsilon)$.

- Next consider the second type of error in $R_\epsilon^{\text{rej}}(f_\delta, \alpha)$: $\max_{\mathbf{x}'' \in \mathcal{N}(\mathbf{x}, \epsilon)} \mathbf{1}\left[f_\delta(\mathbf{x}'') \notin \{y, \perp\}\right] = 1$. This means there exists an $\mathbf{x}'' \in \mathcal{N}(\mathbf{x}, \epsilon)$ such that $f_\delta(\mathbf{x}'') \notin \{y, \perp\}$, *i.e.*, $\mathbf{x}'' \notin \mathcal{N}(f^*, \delta\epsilon)$ and $f^*(\mathbf{x}'') \neq y$. This implies that *all* $\mathbf{z} \in \mathcal{N}(\mathbf{x}'', \delta\epsilon)$ should have $f^*(\mathbf{z}) = f^*(\mathbf{x}'') \neq y$. In particular, there exists $\mathbf{z} \in \mathcal{N}(\mathbf{x}'', \delta\epsilon)$ with $d(\mathbf{z}, \mathbf{x}) \leq (1-\delta)\epsilon$ and $f^*(\mathbf{z}) \neq y$. It can be verified that $\mathbf{z} = \delta\mathbf{x} + (1-\delta)\mathbf{x}''$, which is a point on the line joining $\mathbf{x}$ and $\mathbf{x}''$, satisfies the above condition. In summary, either $f^*(\mathbf{x}) \neq y$, or $f^*(\mathbf{x}) = y \neq f^*(\mathbf{z})$ and thus $\mathbf{x} \in \mathcal{N}(f^*, (1-\delta)\epsilon)$.

Overall, a sample $(\mathbf{x}, y)$ contributing error to $R_\epsilon^{\text{rej}}(f_\delta, \alpha)$ must satisfy either $f^*(\mathbf{x}) \neq y$ or $\mathbf{x} \in \mathcal{N}(f^*, \epsilon')$. Clearly, such a sample also contributes an error to $R_{\epsilon'}(f^*)$. Therefore, we have

$$R_\epsilon^{\text{rej}}(f_\delta, \alpha) \;\leq\; R_{\epsilon'}(f^*), \forall \alpha \in [0,1]. \tag{18}$$

To show that the bound is tight, consider the following data distribution. Let $\mathbf{x} \in \mathbb{R}$ and $y \in \{-1, +1\}$, $\alpha \in [0,1]$, and let $\beta \in (0, \frac{1}{2})$ be some constant. Let the distribution be: $(\mathbf{x}, y)$ is $(-4\epsilon, -1)$ with probability $\frac{1-\beta}{2}$, $(-\frac{\alpha\epsilon}{4}, -1)$ with probability $\frac{\beta}{2}$, $(\frac{\alpha\epsilon}{4}, +1)$ with probability $\frac{\beta}{2}$, and $(4\epsilon, +1)$ with probability $\frac{1-\beta}{2}$. Let $\delta = \frac{1-\alpha}{2}$, $f^*(\mathbf{x}) := \text{sign}(\mathbf{x} + \epsilon)$. It is clear that $R_{\epsilon'}(f^*) = R_{(1+\alpha)\epsilon/2}(f^*) = \frac{\beta}{2}$. It is also clear that any f must have $R_\epsilon^{\text{rej}}(f, \alpha) \geq \frac{\beta}{2}$ since the points $-\frac{\alpha\epsilon}{4}$ and $\frac{\alpha\epsilon}{4}$ have distance only $\frac{\alpha\epsilon}{2}$ but have different labels. □

We note that the proof generalizes that of Theorem 5 in (Tramèr, 2022). In particular, our theorem includes the latter as a special case (corresponding to $\alpha = 0$ and $\delta = \frac{1}{2}$).

B.4. Comparing the Robustness Curves and Total Robust Losses of f_δ and f^* in Theorem 1

Theorem 1 investigates the feasibility of a good selective classifier (w.r.t. the robustness curve and the total robust loss), by constructing a classifier f_δ. It is meaningful to perform a fine-grained analysis into when f_δ has a better total robust loss than the classifier f^* without rejection.

For simplicity, we assume f^* has a 0 standard error on the clean data distribution. The analysis for the case with a nonzero standard error is similar.

Theorem 3. *Consider binary classification. Let $f^*(\mathbf{x})$ be any classifier without a rejection option that has 0 standard error on the data distribution. Suppose the data density for data points with a distance of r to the decision boundary of f^* is $p(r)$. Consider the selective classifier f_δ defined in Theorem 1:*

$$f_\delta(\mathbf{x}) \;:=\; \begin{cases} \perp & \textit{if } \mathbf{x} \in \mathcal{N}(f^*, \delta\epsilon), \\ f^*(\mathbf{x}) & \textit{otherwise.} \end{cases} \tag{19}$$

Then for $\delta \in [0, \frac{1}{2}]$, the total robust loss of f_δ with respect to the rejection loss ℓ^{rej} can be computed by:

$$L_\epsilon(f_\delta; \ell^{rej}) = \int_0^{(1-\delta)\epsilon} p(r)\mathrm{d}r + \int_{(1-\delta)\epsilon}^{(1+\delta)\epsilon} \ell^{rej}(r - \delta\epsilon)\, p(r)\mathrm{d}r. \tag{20}$$

Also, the curve of robust error of f_δ can be computed by:

$$R_\epsilon^{rej}(f_\delta, \alpha) = \begin{cases} \int_0^{(1-\delta)\epsilon} p(r)\mathrm{d}r & \textit{if } \alpha \in [0, 1-2\delta], \\ \int_0^{(\alpha+\delta)\epsilon} p(r)\mathrm{d}r & \textit{if } \alpha \in (1-2\delta, 1]. \end{cases} \tag{21}$$

Proof. First consider the total robust loss. When $0 < r \leq (1-\delta)\epsilon$, the data point will contribute a loss of 1 to the total robust loss. When $(1-\delta)\epsilon < r \leq (1+\delta)\epsilon$, the data point will contribute a loss of $\ell^{\text{rej}}(r - \delta\epsilon)$ to the total robust loss. When $r > (1+\delta)\epsilon$, the data point will contribute 0 loss to the total robust loss. Thus, we have Eq. (20).

For the curve of robust error, the data points with $0 < r \leq (1-\delta)\epsilon$ will always contribute a loss of 1. When $\alpha \in [0, 1-2\delta]$, no data points with $r \geq (1-\delta)\epsilon$ can contribute a loss, either by small perturbations to get a rejection or mis-classification, or by large perturbations to get a mis-classification. When $\alpha \in (1-2\delta, 1]$, data points with $r \in [(1-\delta)\epsilon, (\alpha+\delta)\epsilon]$ will contribute a loss of 1 by small perturbations to get a rejection. Thus, we have Eq. (21). □

Now we are ready to compare f_δ to f^*.

First consider the curve of robust error. It is easy to know that the curve of robust error of f^* is $R_\epsilon^{\text{rej}}(f^*, \alpha) = \int_0^\epsilon p(r)\mathrm{d}r$ for $\alpha \in [0, 1]$. When $\alpha \in [0, 1-\delta]$, we have $R_\epsilon^{\text{rej}}(f^*, \alpha) \geq R_\epsilon^{\text{rej}}(f_\delta, \alpha)$; when $\alpha \in (1-\delta, 1]$, we have $R_\epsilon^{\text{rej}}(f^*, \alpha) \leq R_\epsilon^{\text{rej}}(f_\delta, \alpha)$. When $\alpha \in [0, 1-2\delta]$, if $\int_{(1-\delta)\epsilon}^\epsilon p(r)\mathrm{d}r$ is large, then f_δ will have much lower robust error with rejection than f^*, since $R_\epsilon^{\text{rej}}(f^*, \alpha) - R_\epsilon^{\text{rej}}(f_\delta, \alpha) = \int_{(1-\delta)\epsilon}^\epsilon p(r)\mathrm{d}r$; when $\alpha \in (1-2\delta, 1-\delta]$, if $\int_{(\alpha+\delta)\epsilon}^\epsilon p(r)\mathrm{d}r$ is large, then f_δ will also have much lower robust error with rejection than f^*, since $R_\epsilon^{\text{rej}}(f^*, \alpha) - R_\epsilon^{\text{rej}}(f_\delta, \alpha) = \int_{(\alpha+\delta)\epsilon}^\epsilon p(r)\mathrm{d}r$.

Now consider the total robust loss. The total robust loss of f^* is $\int_0^\epsilon p(r)\mathrm{d}r$, which is larger than that of f_δ by $\int_{(1-\delta)\epsilon}^\epsilon p(r)\mathrm{d}r - \int_{(1-\delta)\epsilon}^{(1+\delta)\epsilon} \ell^{\text{rej}}(r-\delta\epsilon)\, p(r)\mathrm{d}r$. More precisely, both f^* and f_δ will get mis-classification loss on some perturbations of points with distance in the range $[0, (1-\delta)\epsilon]$ from the decision boundary of f^*. This is because there always exist some large perturbations of these points crossing the boundary. On the other hand, f^* simply gets mis-classification loss from perturbations of the points with distance in the range $[(1-\delta)\epsilon, \epsilon]$. While for all points with distance larger than $(1-\delta)\epsilon$, f_δ can correctly classify all their small perturbations of magnitude at most $(1-2\delta)\epsilon$, and reject or correctly classify their larger perturbations. So it only gets rejection loss for large magnitudes, which can then potentially lead to smaller total robust losses than f^* for monotonically non-increasing rejection losses.

Therefore, for some data distributions, there exists a fixed δ such that f_δ can get small total robust losses with respect to a wide range of reasonable rejection losses. For example, consider the step rejection losses $\ell^{\text{rej}}(r) = \mathbf{1}\{r \leq \alpha_0\epsilon\}$ with parameter in the range $\alpha_0 \in [0, \bar{\alpha}_0]$ where $\bar{\alpha}_0 \in [0, 1]$. If we set $\delta = \frac{1-\bar{\alpha}_0}{2}$, then $1 - 2\delta = \bar{\alpha}_0$ and $\alpha \in [0, 1-2\delta]$. The

total robust loss of f_δ with respect to these rejection losses is $\int_0^{\frac{(1+\bar{\alpha}_0)}{2}\epsilon} p(r)\mathrm{d}r$. In contrast, the total robust loss of f^* is $\int_0^{\epsilon} p(r)\mathrm{d}r$, which can be significantly larger than that of f_δ. The total robust loss of f_δ is smaller than that of f^* by the amount $\int_{\frac{(1+\bar{\alpha}_0)}{2}\epsilon}^{\epsilon} p(r)\mathrm{d}r$. If the probability mass of points with $r \in [\frac{(1+\bar{\alpha}_0)}{2}\epsilon, \epsilon]$ is large, then the improvement is significant.

C. Calculating the Total Robust Loss

In this section, we discuss the calculation of the total robust loss for specific instantiations of the rejection loss ℓ^{rej}. Given the curve of robust error $\{s(\alpha) := R_\epsilon^{\text{rej}}(f, \alpha) \ : \ \alpha \in [0, 1]\}$ and a specific choice of rejection loss $\ell^{\text{rej}}(r)$ that is monotonically non-increasing, we can use Eq. (15) from Lemma 2 to calculate the total robust loss:

$$L_\epsilon(f; \ell^{\text{rej}}) \ = \ R_\epsilon^{\text{rej}}(f, 0) \ + \ (\ell^{\text{rej}}(0) \ - \ 1)\, p_{\text{rej}} \ + \ \int_0^1 \ell^{\text{rej}}(\alpha\epsilon)\, \mathrm{d}s(\alpha). \tag{22}$$

As discussed in Corollary 1, let us additionally choose the rejection loss to be differentiable almost everywhere, and satisfy the conditions $\ell^{\text{rej}}(0) = 1$ and $\ell^{\text{rej}}(\epsilon) = 0$. This is satisfied *e.g.*, by the ramp and step rejection losses defined in Eqs. (4) and (3). Applying the product rule (or integration by parts), the integral term in the total robust loss can be expressed as

$$\begin{aligned}\int_0^1 \ell^{\text{rej}}(\alpha\epsilon)\, \mathrm{d}s(\alpha) \ &= \ \ell^{\text{rej}}(\epsilon)\, s(1) \ - \ \ell^{\text{rej}}(0)\, s(0) \ - \ \int_0^1 s(\alpha)\, \mathrm{d}\ell^{\text{rej}}(\alpha\epsilon) \\ &= \ -\, s(0) \ - \ \int_0^1 s(\alpha)\, \mathrm{d}\ell^{\text{rej}}(\alpha\epsilon).\end{aligned}$$

Substituting the above into Eq. (22), the total robust loss simplifies into

$$L_\epsilon(f; \ell^{\text{rej}}) \ = \ -\int_0^1 s(\alpha)\, \mathrm{d}\ell^{\text{rej}}(\alpha\epsilon). \tag{23}$$

From the above expression, we next calculate the total robust loss for the ramp and step rejection losses.

C.1. Ramp Rejection loss

Recall that the ramp rejection loss is defined as

$$\ell^{\text{rej}}(r) \ = \ \left(1 \ - \ \frac{r}{\epsilon}\right)^t, \ \ r \in [0, \epsilon]$$

for some $t \geq 1$. We have $\ell^{\text{rej}}(\alpha\epsilon) \ = \ (1 - \alpha)^t$ and

$$\mathrm{d}\ell^{\text{rej}}(\alpha\epsilon) \ = \ -t\,(1 \ - \ \alpha)^{t-1}\, \mathrm{d}\alpha.$$

Substituting the above into Eq. (23) gives the total robust loss

$$L_\epsilon(f; \ell^{\text{rej}}) \ = \ t \int_0^1 s(\alpha)\,(1 \ - \ \alpha)^{t-1}\, \mathrm{d}\alpha.$$

For the special case $t = 1$, this reduces to the area under the robust error curve $\int_0^1 s(\alpha)\, \mathrm{d}\alpha$.
For the special case $t = 2$, this reduces to $2\int_0^1 s(\alpha)\,(1 - \alpha)\, \mathrm{d}\alpha$ (and so on for larger t).
In our experiments, we calculate the total robust loss for $t \in \{1, 2, 3, 4\}$. Since we calculate the robust error curve only for a finite set of α values, we approximate the above integrals using the trapezoidal rule. We use finely-spaced α values in $[0, 1]$ with a spacing of 0.01, and use linear interpolation to obtain intermediate (missing) values of the robust error curve.

C.2. Step Rejection loss

Recall that the step rejection loss is defined as

$$\ell^{\text{rej}}(r) \ = \ \mathbf{1}\{r \ \leq \ \alpha_0\epsilon\}, \ \ r \in [0, \epsilon]$$

for some $\alpha_0 \in [0, 1]$. In this case, there are two ways to calculate the total robust loss, both leading to the same result.

Approach 1 For the step rejection loss, the derivative can be defined by appealing to the Dirac delta function $\delta(x)$. Specifically, $\ell^{\text{rej}}(\alpha\epsilon) = \mathbf{1}\{\alpha \leq \alpha_0\}$ and

$$\mathrm{d}\ell^{\text{rej}}(\alpha\epsilon) \;=\; -\,\delta(\alpha - \alpha_0)\,\mathrm{d}\alpha,$$

where the negative sign arises because the step loss drops from 1 to 0 at $\alpha = \alpha_0$. Substituting the above into Eq. (23) gives the total robust loss

$$L_\epsilon(f; \ell^{\text{rej}}) \;=\; \int_0^1 s(\alpha)\,\delta(\alpha - \alpha_0)\,\mathrm{d}\alpha \;=\; s(\alpha_0) \;=\; R_\epsilon^{\text{rej}}(f, \alpha_0).$$

Approach 2 We directly use the definition of the total robust loss in Eq. (22), which for the step rejection loss is

$$\begin{aligned} L_\epsilon(f; \ell^{\text{rej}}) \;&=\; R_\epsilon^{\text{rej}}(f, 0) \;+\; \int_0^1 \mathbf{1}\{\alpha \leq \alpha_0\}\,\mathrm{d}s(\alpha) \\ &=\; s(0) \;+\; \int_0^{\alpha_0} 1\,\mathrm{d}s(\alpha) \;=\; s(0) \;+\; s(\alpha_0) \;-\; s(0) \\ &=\; s(\alpha_0) \;=\; R_\epsilon^{\text{rej}}(f, \alpha_0). \end{aligned}$$

For the step rejection loss, the total robust loss is equal to the value of the robust error curve at the point $\alpha = \alpha_0$.

D. Experimental Setup

We ran all our experiments with PyTorch and NVDIA GeForce RTX 2080Ti GPUs. We ran all the experiments once with fixed random seeds. The code for our work can be found at `https://github.com/jfc43/stratified-adv-rej`.

D.1. Datasets

MNIST. MNIST (LeCun, 1998) is a large dataset of handwritten digits with 10 classes. Each digit has 5,500 training images and 1,000 test images. Each image is 28×28 grayscale. We normalize the range of pixel values to [0,1].

SVHN. SVHN (Netzer et al., 2011) is a real-world image dataset of 32×32 color images with 10 classes (one for each digit). It is obtained from house numbers in Google Street View images. The training set has 73,257 images and the original test set has 26,032 images. We use the first 10,000 images from the original test set as the test set for our experiments. We normalize the range of pixel values to [0,1].

CIFAR-10. CIFAR-10 (Krizhevsky et al., 2009) is a dataset of 32×32 color images with ten classes, each consisting of 5,000 training images and 1,000 test images. The classes correspond to categories such as dogs, frogs, ships, trucks, etc. We normalize the range of pixel values to [0,1].

D.2. Baseline Methods

We consider the following five baseline methods: (1) AT+CR: adversarial training (AT) (Madry et al., 2018) with Confidence-based Rejection; (2)TRADES+CR: TRADES (Zhang et al., 2019) with Confidence-based Rejection; (3) CCAT: confidence-calibrated adversarial training (Stutz et al., 2020); (4) RCD: robust classification with detection (Sheikholeslami et al., 2021); (5) ATRR: adversarial training with rectified rejection (Pang et al., 2022). We provide their training details below.

AT+CR. We use the standard adversarial training (AT) proposed in (Madry et al., 2018) to train the base model. On MNIST, we use LetNet network architecture (LeCun et al., 1989) and train the network for 100 epochs with a batch size of 128. We use standard stochastic gradient descent (SGD) starting with a learning rate of 0.1. The learning rate is multiplied by 0.95 after each epoch. We use a momentum of 0.9 and do not use weight decay for SGD. We use the PGD attack to generate adversarial training examples with $\epsilon = 0.3$, a step size of 0.01, 40 steps and a random start. On SVHN, we use ResNet-20 network architecture (He et al., 2016) and train the network for 200 epochs with a batch size of 128. We use standard SGD starting with a learning rate of 0.1. The learning rate is multiplied by 0.95 after each epoch. We use a momentum of 0.9 and do not use weight decay for SGD. We use the PGD attack to generate adversarial training examples with $\epsilon = \frac{8}{255}$, a step size

of $\frac{2}{255}$, 10 steps and a random start. On CIFAR-10, we use ResNet-20 network architecture and train the network following the suggestions in (Pang et al., 2021). Specifically, we train the network for 110 epochs with a batch size of 128 using SGD with Nesterov momentum and learning rate schedule. We set momentum 0.9 and ℓ_2 weight decay with a coefficient of 5×10^{-4}. The initial learning rate is 0.1 and it decreases by 0.1 at 100 and 105 epoch respectively. We augment the training images using random crop and random horizontal flip. We use the PGD attack to generate adversarial training examples with $\epsilon = \frac{8}{255}$, a step size of $\frac{2}{255}$, 10 steps and a random start.

TRADES+CR. TRADES is an adversarial training method proposed in (Zhang et al., 2019). We follow their original setup to train the models on MNIST and CIFAR-10. The setup for SVHN is the same as that for CIFAR-10.

CCAT. We follow the original training settings for CCAT in (Stutz et al., 2020) and train models on MNIST, SVHN and CIFAR-10 using standard stochastic gradient descent (SGD). On all datasets, we set $\rho = 10$. On MNIST, we use LetNet network architecture and train the network for 100 epochs with a batch size of 100 and a learning rate of 0.1. On SVHN, we use ResNet-20 network architecture and train the network for 200 epochs with a batch size of 100 and a learning rate of 0.1. On CIFAR-10, we use ResNet-20 network architecture and train the network for 200 epochs with a batch size of 100 and a learning rate of 0.075. We augment the training images using random crop and random horizontal flip. On all datasets, we use learning rate schedule and the learning rate is multiplied by 0.95 after each epoch. We use a momentum of 0.9 and do not use weight decay for SGD. We use the PGD attack with backtracking to generate adversarial training examples: we use a learning rate of 0.005, a momentum of 0.9, a learning rate factor of 1.5, 40 steps and a random start. We randomly switch between random initialization and zero initialization. We train on 50% clean and 50% adversarial examples per batch.

RCD. We consider a training loss proposed in (Sheikholeslami et al., 2021) (see their Equation 14) to train a neural network for (k+1)-class classification with the (k+1)-th class dedicated to the detection task. (Sheikholeslami et al., 2021) use interval bound propagation (IBP) technique (Gowal et al., 2019) to bound the training loss and then train the network by minimizing the tractable upper bound on the training loss (see their Equation 18) to achieve verified robustness. Since we don't consider verified robustness in our paper, we use the PGD attack to solve the inner maximization problems of the training objective instead. For readers' convenience, we describe their training objective here. Suppose the logits of the network is $\widetilde{\mathbf{h}}(\mathbf{x}\,;\boldsymbol{\theta}) = [\widetilde{h}_1(\mathbf{x}\,;\boldsymbol{\theta}), \cdots, \widetilde{h}_{k+1}(\mathbf{x}\,;\boldsymbol{\theta})]$ and the softmax output of the network is $\mathbf{h}(\mathbf{x}\,;\boldsymbol{\theta}) = [h_1(\mathbf{x}\,;\boldsymbol{\theta}), \cdots, h_{k+1}(\mathbf{x}\,;\boldsymbol{\theta})]$. The softmax output of the network is obtained by applying the softmax function to the logits. Then the training objective is

$$\min_{\boldsymbol{\theta}} \; \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \left[\max_{\mathbf{x}' \in \mathcal{N}(\mathbf{x},\epsilon)} -\log[h_y(\mathbf{x}'\,;\boldsymbol{\theta})] \; + \; \lambda_1 \, L^{\text{abstain}}_{\text{robust}}(\mathbf{x}, y; \boldsymbol{\theta}) \; - \; \lambda_2 \, \log[h_y(\mathbf{x}\,;\boldsymbol{\theta})] \right]. \tag{24}$$

where

$$L^{\text{abstain}}_{\text{robust}}(\mathbf{x}, y; \boldsymbol{\theta}) = \max_{\mathbf{x}' \in \mathcal{N}(\mathbf{x},\epsilon)} \min \left\{ -\log(\frac{e^{\widetilde{h}_y(\mathbf{x}'\,;\boldsymbol{\theta})}}{\sum_{i\in\mathcal{I}\setminus\{k+1\}} e^{\widetilde{h}_i(\mathbf{x}'\,;\boldsymbol{\theta})}}), -\log(\frac{e^{\widetilde{h}_{k+1}(\mathbf{x}'\,;\boldsymbol{\theta})}}{\sum_{i\in\mathcal{I}\setminus\{y\}} e^{\widetilde{h}_i(\mathbf{x}'\,;\boldsymbol{\theta})}}) \right\}, \tag{25}$$

and $\mathcal{I} = \{1, 2, \ldots, k+1\}$. (Sheikholeslami et al., 2021) always set $\lambda_1 = 1$ and to keep high clean accuracy, they set a larger λ_2 (e.g. they set $\lambda_2 = 2$ on MNIST and set $\lambda_2 = 2.9$ on CIFAR-10) since it is hard to get high clean accuracy when using the IBP technique to train models. Since we don't use the IBP technique, we simply set $\lambda_1 = 1$ and $\lambda_2 = 1$ (our results show that setting $\lambda_2 = 1$ leads to better results than that of setting $\lambda_2 \geq 2$). We train models on MNIST, SVHN, and CIFAR-10 using standard stochastic gradient descent (SGD). We split each training batch into two sub-batches of equal size, and use the first sub-batch for the last two loss terms in the training objective and use the second sub-batch for the first loss term in the training objective. On MNIST, we use LetNet network architecture and train the network for 100 epochs with a batch size of 128. We use standard SGD starting with a learning rate of 0.1. The learning rate is multiplied by 0.95 after each epoch. We use a momentum of 0.9 and do not use weight decay for SGD. We use the PGD attack to solve the inner maximization problems with $\epsilon = 0.3$, a step size of 0.01, 40 steps and a random start. On SVHN, we use ResNet-20 network architecture and train the network for 200 epochs with a batch size of 128. We use standard SGD starting with a learning rate of 0.05. The learning rate is multiplied by 0.95 after each epoch. We use a momentum of 0.9 and do not use weightf decay for SGD. We use the PGD attack to solve the inner maximization problems with $\epsilon = \frac{8}{255}$, a step size of $\frac{2}{255}$, 10 steps and a random start. On CIFAR-10, we use ResNet-20 network architecture and train the network for 110 epochs with a batch size of 128 using SGD with Nesterov momentum and learning rate schedule. We set momentum 0.9 and ℓ_2 weight decay with a coefficient of 5×10^{-4}. The initial learning rate is 0.1 and it decreases by 0.1 at 100 and 105 epoch respectively. We augment the training images using random crop and random horizontal flip. We use the PGD attack to solve the inner maximization problems with $\epsilon = \frac{8}{255}$, a step size of $\frac{2}{255}$, 10 steps and a random start.

ATRR. We follow the original training settings for ATRR in (Pang et al., 2022) and train models on MNIST, SVHN and CIFAR-10 using standard stochastic gradient descent (SGD). On MNIST, we use LetNet network architecture and train the network for 100 epochs with a batch size of 128. We use standard SGD starting with a learning rate of 0.1. The learning rate is multiplied by 0.95 after each epoch. We use a momentum of 0.9 and do not use weight decay for SGD. We use the PGD attack to generate adversarial training examples with $\epsilon = 0.3$, a step size of 0.01, 40 steps and a random start. On SVHN, we use ResNet-20 network architecture and train the network for 200 epochs with a batch size of 128. We use standard SGD starting with a learning rate of 0.1. The learning rate is multiplied by 0.95 after each epoch. We use a momentum of 0.9 and do not use weight decay for SGD. We use the PGD attack to generate adversarial training examples with $\epsilon = \frac{8}{255}$, a step size of $\frac{2}{255}$, 10 steps and a random start. On CIFAR-10, we use ResNet-20 network architecture and train the network for 110 epochs with a batch size of 128 using SGD with Nesterov momentum and learning rate schedule. We set momentum 0.9 and ℓ_2 weight decay with a coefficient of 5×10^{-4}. The initial learning rate is 0.1 and it decreases by 0.1 at 100 and 105 epoch respectively. We augment the training images using random crop and random horizontal flip. We use the PGD attack to generate adversarial training examples with $\epsilon = \frac{8}{255}$, a step size of $\frac{2}{255}$, 10 steps and a random start.

E. Designing Adaptive Attacks

To compute the robust accuracy with rejection $A_\epsilon^{\text{rej}}(f, \alpha)$ for a given $\epsilon > 0$ and $\alpha \in [0, 1]$, we need to generate two adversarial examples $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)$ and $\mathbf{x}'' \in \mathcal{N}(\mathbf{x}, \epsilon)$ for each clean input $(\mathbf{x}, y)$. We call the attack for generating $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)$ the inner attack, and the attack for generating $\mathbf{x}'' \in \mathcal{N}(\mathbf{x}, \epsilon)$ the outer attack.

For both the inner attack and outer attack, we use an ensemble of attacks and report the worst-case robustness. For the inner attack, if any of the attacks in the inner-attack ensemble finds an adversarial example $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)$ that is rejected by the model, then we consider the inner attack to be successful on the clean input $(\mathbf{x}, y)$. For the outer attack, if any of the attacks in the outer-attack ensemble finds an adversarial example $\mathbf{x}'' \in \mathcal{N}(\mathbf{x}, \epsilon)$ that is accepted and misclassified by the model, then we consider the outer attack to be successful on the clean input $(\mathbf{x}, y)$.

In this section, we design adaptive attacks to generate $\mathbf{x}'$ and $\mathbf{x}''$ for the different methods using the same underlying principles to ensure fair comparison. The following ideas are applied to design the attack loss functions for all the methods. Whenever the attack objective is to maximize a probability (or a probability-like) term $p \in [0, 1]$, we use the loss function $\log(p) \in (-\infty, 0]$. Similarly, the loss function $-\log(p) \in [0, \infty)$ is used to minimize a probability (or a probability-like) term p.

We first introduce the attack objectives for all the methods below, and then discuss how we solve the attack objectives.

E.1. Attack Objectives

In this section, we describe the attack objectives used to generate the adversarial examples $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)$ (*i.e.*, the inner attack) and the adversarial examples $\mathbf{x}'' \in \mathcal{N}(\mathbf{x}, \epsilon)$ (*i.e.*, the outer attack) from a clean input $(\mathbf{x}, y)$ for the different methods compared. The goal of the adversary to generate $\mathbf{x}'$ is to make the defense method reject $\mathbf{x}'$. The goal of the adversary to generate $\mathbf{x}''$ is to make the defense method accept and incorrectly classify $\mathbf{x}''$ into a class other than y. We next discuss the inner and outer attack objectives for the different methods considered.

Confidence-based Rejection. The methods AT+CR, TRADES+CR and CCAT (Stutz et al., 2020) use the classifier's confidence (*i.e.*, maximum softmax probability) as the score for rejection. Suppose the softmax output of the classifier is $\mathbf{h}(\mathbf{x}\,;\boldsymbol{\theta}) = [h_1(\mathbf{x}\,;\boldsymbol{\theta}), \cdots, h_k(\mathbf{x}\,;\boldsymbol{\theta})]$, then the score for acceptance is the confidence given by $h_{\max}(\mathbf{x}\,;\boldsymbol{\theta}) = \max_j h_j(\mathbf{x}\,;\boldsymbol{\theta})$. We use the log-sum-exp approximation of the max function to define a smooth inner attack objective that minimizes the confidence score $h_{\max}(\mathbf{x}\,;\boldsymbol{\theta})$. We use the fact that

$$\frac{1}{\tau} \log\Big(\sum_{i=1}^{k} e^{\tau s_i}\Big) \;\approx\; \max_{i\in[k]} s_i$$

where the approximation becomes better for larger values of the temperature constant $\tau > 0$. We would like to approximate the exact inner attack objective given by

$$\mathbf{x}' \;=\; \operatorname*{argmax}_{\mathbf{z}\in\mathcal{N}(\mathbf{x},\alpha\epsilon)} \; -\log h_{\max}(\mathbf{z}\,;\boldsymbol{\theta}). \tag{26}$$

This attack aims to find an adversarial input $\mathbf{x}'$ that minimizes the confidence, thus causing the input to be rejected by

methods using confidence-based rejection. Let $\widetilde{\mathbf{h}}(\mathbf{x}\,;\boldsymbol{\theta}) = [\widetilde{h}_1(\mathbf{x}\,;\boldsymbol{\theta}), \cdots, \widetilde{h}_k(\mathbf{x}\,;\boldsymbol{\theta})]$ denote the logits corresponding to the classifier prediction, with $\widetilde{h}_{\max}(\mathbf{x}\,;\boldsymbol{\theta})$ being the maximum logit. The attack objective (26) can be approximated as

$$\begin{aligned}
\mathbf{x}' &= \operatorname*{argmax}_{\mathbf{z}\in\mathcal{N}(\mathbf{x},\alpha\epsilon)} -\log h_{\max}(\mathbf{z}\,;\boldsymbol{\theta}) \\
&= \operatorname*{argmax}_{\mathbf{z}\in\mathcal{N}(\mathbf{x},\alpha\epsilon)} -\widetilde{h}_{\max}(\mathbf{z}\,;\boldsymbol{\theta}) \;+\; \log\Big(\sum_{i=1}^{k} e^{\widetilde{h}_i(\mathbf{z}\,;\boldsymbol{\theta})}\Big) \\
&\approx \operatorname*{argmax}_{\mathbf{z}\in\mathcal{N}(\mathbf{x},\alpha\epsilon)} -\frac{1}{\tau}\log\Big(\sum_{i=1}^{k} e^{\tau\widetilde{h}_i(\mathbf{z}\,;\boldsymbol{\theta})}\Big) \;+\; \log\Big(\sum_{i=1}^{k} e^{\widetilde{h}_i(\mathbf{z}\,;\boldsymbol{\theta})}\Big).
\end{aligned} \tag{27}$$

We name this attack with objective (27) Low Confidence Inner Attack (**LCIA**). In our experiments, we set $\tau = 100$.

For the outer attack, we use the multi-targeted PGD approach. Specifically, for each target label $j \neq y$, we generate an adversarial example $\mathbf{x}''_j \in \mathcal{N}(\mathbf{x}, \epsilon)$ via the following objective:

$$\mathbf{x}''_j \;=\; \operatorname*{argmax}_{\mathbf{z}\in\mathcal{N}(\mathbf{x},\epsilon)} \log h_j(\mathbf{z}\,;\boldsymbol{\theta}). \tag{28}$$

Then we select the strongest adversarial example $\mathbf{x}''$ via:

$$\mathbf{x}'' \;=\; \mathbf{x}''_{j^\star} \quad s.t. \quad j^\star \;=\; \operatorname*{argmax}_{j\in[k]\setminus\{y\}} \log h_j(\mathbf{x}''_j\,;\boldsymbol{\theta}). \tag{29}$$

By solving this objective, the adversary attempts to find an adversarial example that is misclassified with high confidence. The goal of the adversary is to make the selective classifier accept and incorrectly classify the adversarial input. We name this attack with objective (28) High Confidence Misclassification Outer Attack (**HCMOA**). Note that this HCMOA attack is stronger than the PGD attack with backtracking proposed in (Stutz et al., 2020) for evaluating the robustness of CCAT (see Appendix F.7 for a comparison).

RCD. The RCD method (Sheikholeslami et al., 2021) trains a $(k+1)$-way classifier such that class $k+1$ is treated as the rejection class. Suppose the softmax output of the classifier is $\mathbf{h}(\mathbf{x}\,;\boldsymbol{\theta}) = [h_1(\mathbf{x}\,;\boldsymbol{\theta}), \cdots, h_{k+1}(\mathbf{x}\,;\boldsymbol{\theta})]$. For the inner attack, we generate the adversarial example $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)$ using the following objective:

$$\mathbf{x}' \;=\; \operatorname*{argmax}_{\mathbf{z}\in\mathcal{N}(\mathbf{x},\alpha\epsilon)} \log h_{k+1}(\mathbf{z}\,;\boldsymbol{\theta}). \tag{30}$$

The goal of the adversary is to make the method reject the adversarial input by pushing the probability of class $k+1$ close to 1. We name this attack with objective (30) RCD Inner Attack (**RCDIA**).

For the outer attack, we use the multi-targeted PGD approach. Specifically, for each target label $j \notin \{y, k+1\}$, we generate the adversarial example $\mathbf{x}''_j \in \mathcal{N}(\mathbf{x}, \epsilon)$ via the following objective:

$$\mathbf{x}''_j \;=\; \operatorname*{argmax}_{\mathbf{z}\in\mathcal{N}(\mathbf{x},\epsilon)} \log h_j(\mathbf{z}\,;\boldsymbol{\theta}). \tag{31}$$

Then we select the strongest adversarial example $\mathbf{x}''$ via:

$$\mathbf{x}'' \;=\; \mathbf{x}''_{j^\star} \quad s.t. \quad j^\star \;=\; \operatorname*{argmax}_{j\notin\{y,k+1\}} \log h_j(\mathbf{x}''_j\,;\boldsymbol{\theta}). \tag{32}$$

Here, the goal of the adversary is to make the selective classifier accept and incorrectly classify the adversarial input, and this objective achieves this by increasing the probability of a class that is different from both the true class y and the rejection class $k+1$ to be close to 1. We name this attack with objective (31) RCD Outer Attack (**RCDOA**).

ATRR. The ATRR method (Pang et al., 2022) uses a rectified confidence score for rejection. Suppose the softmax output of the classifier is $\mathbf{h}(\mathbf{x}\,;\boldsymbol{\theta}) = [h_1(\mathbf{x}\,;\boldsymbol{\theta}), \cdots, h_k(\mathbf{x}\,;\boldsymbol{\theta})]$ and the auxiliary function is $A(\mathbf{x};\phi) \in [0,1]$. The rectified confidence is defined by Pang et al. (2022) as the product of the auxiliary function and the classifier's confidence,

i.e., $A(\mathbf{x};\phi)\, h_{\max}(\mathbf{x}\,;\boldsymbol{\theta})$. For the inner attack, we generate the adversarial example $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)$ using the following objective:

$$\begin{aligned}\mathbf{x}' \;=\; & \underset{\mathbf{z}\in\mathcal{N}(\mathbf{x},\alpha\epsilon)}{\operatorname{argmax}} \; -\log A(\mathbf{z};\phi) \;-\; \log h_{\max}(\mathbf{z}\,;\boldsymbol{\theta}) \\ \approx\; & \underset{\mathbf{z}\in\mathcal{N}(\mathbf{x},\alpha\epsilon)}{\operatorname{argmax}} \; -\log A(\mathbf{z};\phi) \;+\; \Big[\log\Big(\sum_{i=1}^{k} e^{\widetilde{h}_i(\mathbf{z}\,;\boldsymbol{\theta})}\Big) - \frac{1}{\tau}\,\log\Big(\sum_{i=1}^{k} e^{\tau\widetilde{h}_i(\mathbf{z}\,;\boldsymbol{\theta})}\Big)\Big]. \end{aligned} \tag{33}$$

Here, we use the log-sum-exp approximation of the max function $h_{\max}(\mathbf{z}\,;\boldsymbol{\theta}) = \max_j h_j(\mathbf{z}\,;\boldsymbol{\theta})$ and set $\tau = 100$. The goal is to minimize the rectified confidence by either minimizing the auxiliary function (first term) or the classifier's confidence by pushing its predictions to be close to uniform (second term). By minimizing the rectified confidence score, the adversary attempts to make the ATRR method reject the perturbed input $\mathbf{x}'$. We name this attack with objective (33) ATRR Inner Attack (**ATRRIA**).

For the outer attack, we use the multi-targeted PGD approach. Specifically, for each target label $j \neq y$, we generate the adversarial example $\mathbf{x}''_j \in \mathcal{N}(\mathbf{x}, \epsilon)$ via the following objective:

$$\mathbf{x}''_j \;=\; \underset{\mathbf{z}\in\mathcal{N}(\mathbf{x},\epsilon)}{\operatorname{argmax}} \log\Big[A(\mathbf{z};\phi)\, h_j(\mathbf{z}\,;\boldsymbol{\theta})\Big]. \tag{34}$$

Then we select the strongest adversarial example $\mathbf{x}''$ via:

$$\mathbf{x}'' \;=\; \mathbf{x}''_{j^\star} \quad s.t. \quad j^\star \;=\; \underset{j\in[k]\setminus\{y\}}{\operatorname{argmax}} \log\Big[A(\mathbf{x}''_j;\phi)\, h_j(\mathbf{x}''_j\,;\boldsymbol{\theta})\Big]. \tag{35}$$

The goal of the adversary is to make the selective classifier accept and incorrectly classify the adversarial input. The objective achieves this by pushing the rectified confidence as well as the predicted probability of a class different from y close to 1. This ensures that adversarial input is accepted as well as incorrectly classified. We name this attack with objective (34) ATRR Outer Attack (**ATRROA**).

CPR (proposed defense). The goal of the inner attack for CPR is to find $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)$ such that the base model has different predictions on $\mathbf{x}'$ and $T(\mathbf{x}')$, thus ensuring rejection. We consider three adaptive inner attacks that can achieve this goal. The first one is the Low-Confidence Inner Attack (**LCIA**) introduced in Section 4.2. This attack aims to find $\mathbf{x}' \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)$ where the base model has low confidence. Recall that the mapping $T(\mathbf{x}')$ attempts to minimize the base model's probability on the predicted class $\widehat{y}(\mathbf{x}')$. So, if the base model has low confidence on $\mathbf{x}'$, then it will very likely have even lower probability for $\widehat{y}(\mathbf{x}')$ on $T(\mathbf{x}')$, and thus have different predictions on $T(\mathbf{x}')$ and $\mathbf{x}'$.

The second inner attack is a variant of LCIA, named Consistent-Low-Confidence Inner Attack (**CLCIA**), which attempts to find an adversarial example $\mathbf{x}'$ by minimizing the confidence of the base model on both $\mathbf{x}'$ and $T(\mathbf{x}')$. The CLCIA attack has the following objective:

$$\begin{aligned}\mathbf{x}' \;=\; & \underset{\mathbf{z}\in\mathcal{N}(\mathbf{x},\alpha\epsilon)}{\operatorname{argmax}} \Big[-\log h_{\max}(\mathbf{z}\,;\boldsymbol{\theta}) \;-\; \log h_{\max}(T(\mathbf{z})\,;\boldsymbol{\theta})\Big] \\ \approx\; & \underset{\mathbf{z}\in\mathcal{N}(\mathbf{x},\alpha\epsilon)}{\operatorname{argmax}} \Big[-\frac{1}{\tau}\,\log\Big(\sum_{i=1}^{k} e^{\tau\widetilde{h}_i(\mathbf{z}\,;\boldsymbol{\theta})}\Big) \;+\; \log\Big(\sum_{i=1}^{k} e^{\widetilde{h}_i(\mathbf{z}\,;\boldsymbol{\theta})}\Big) \\ & \qquad -\frac{1}{\tau}\,\log\Big(\sum_{i=1}^{k} e^{\tau\widetilde{h}_i(T(\mathbf{z})\,;\boldsymbol{\theta})}\Big) \;+\; \log\Big(\sum_{i=1}^{k} e^{\widetilde{h}_i(T(\mathbf{z})\,;\boldsymbol{\theta})}\Big)\Big]. \end{aligned} \tag{36}$$

We have denoted the logits of the base classifier by $\widetilde{\mathbf{h}}(\mathbf{x}\,;\boldsymbol{\theta}) = [\widetilde{h}_1(\mathbf{x}\,;\boldsymbol{\theta}), \cdots, \widetilde{h}_k(\mathbf{x}\,;\boldsymbol{\theta})]$, and we apply the log-sum-exp approximation to the $\max$ function (as before) with $\tau = 100$. We use the backward pass differentiable approximation (BPDA) method (Athalye et al., 2018) to solve this objective since $T(\mathbf{x})$ does not have a closed-form expression and is not differentiable.

The third inner attack is a multi-targeted attack, also based on BPDA, which considers all possible target classes and attempts to find an $\mathbf{x}'$ such that the base model has high probability for the target class at $\mathbf{x}'$ and a low probability for the target class

at $T(\mathbf{x}')$ (thereby encouraging rejection). The attack objective is: for each target class $j = 1, \ldots, k$,

$$\mathbf{x}'_j \;=\; \underset{\mathbf{z} \in \mathcal{N}(\mathbf{x}, \alpha\epsilon)}{\operatorname{argmax}} \Big[\log h_j(\mathbf{z}\,; \boldsymbol{\theta}) \;-\; \log h_j(T(\mathbf{z})\,; \boldsymbol{\theta}) \Big]. \tag{37}$$

Then we select the strongest adversarial example $\mathbf{x}'$ via:

$$\mathbf{x}' \;=\; \mathbf{x}'_{j^\star} \quad s.t. \quad j^\star \;=\; \underset{j \in [k]}{\operatorname{argmax}} \Big[\log h_j(\mathbf{x}'_j\,; \boldsymbol{\theta}) \;-\; \log h_j(T(\mathbf{x}'_j)\,; \boldsymbol{\theta}) \Big]. \tag{38}$$

We name this third inner attack Prediction-Disagreement Inner Attack (**PDIA**).

Given a clean input $(\mathbf{x}, y)$, the goal of the outer attack is to find $\mathbf{x}'' \in \mathcal{N}(\mathbf{x}, \epsilon)$ such that the base model has consistent wrong predictions on both $\mathbf{x}''$ and $T(\mathbf{x}'')$. This ensures that $\mathbf{x}''$ is accepted and mis-classified. We consider two adaptive outer attacks that can achieve this goal. As discussed in Section 4.2, the first outer attack is a multi-targeted attack based on BPDA with the following objective: for each target class $j \in [k] \setminus \{y\}$,

$$\mathbf{x}''_j \;=\; \underset{\mathbf{z} \in \mathcal{N}(\mathbf{x}, \epsilon)}{\operatorname{argmax}} \Big[\log h_j(\mathbf{z}\,; \boldsymbol{\theta}) \;+\; \log h_j(T(\mathbf{z})\,; \boldsymbol{\theta}) \Big]. \tag{39}$$

Then we select the strongest adversarial example $\mathbf{x}''$ via:

$$\mathbf{x}'' \;=\; \mathbf{x}''_{j^\star} \quad s.t. \quad j^\star \;=\; \underset{j \in [k] \setminus \{y\}}{\operatorname{argmax}} \Big[\log h_j(\mathbf{x}''_j\,; \boldsymbol{\theta}) \;+\; \log h_j(T(\mathbf{x}''_j)\,; \boldsymbol{\theta}) \Big]. \tag{40}$$

We name this outer attack Consistent High Confidence Misclassification Outer Attack (**CHCMOA**).

The second outer attack is the High Confidence Misclassification Outer Attack (**HCMOA**) that was discussed earlier for the methods based on confidence-based rejection. The attack objective is given in Eqns. (28) and (29). The intuition for this attack is that if the base model has a high-confidence incorrect prediction on $\mathbf{x}''$, then it becomes hard for T to change the incorrect prediction.

E.2. Solving the Attack Objectives

We use the PGD with momentum to solve the attack objectives, and use the PGD attack with multiple restarts for evaluating the robustness. Following (Stutz et al., 2020), we initialize the perturbation $\boldsymbol{\delta}$ uniformly over directions and norm as follows:

$$\boldsymbol{\delta} \;=\; u\,\epsilon\,\frac{\boldsymbol{\delta}'}{\|\boldsymbol{\delta}'\|_\infty},\; \boldsymbol{\delta}' \sim \mathcal{N}(\mathbf{0}, \mathbf{I}),\; u \sim U(0,1) \tag{41}$$

where $\boldsymbol{\delta}'$ is sampled from the standard Gaussian and $u \in [0, 1]$ is sampled from the uniform distribution. We also include zero initialization, *i.e.*, $\boldsymbol{\delta} = \mathbf{0}$ as a candidate. We allocate one restart for zero initialization, and multiple restarts for the random initializations. We finally select the perturbation corresponding to the best objective value obtained throughout the optimization.

When solving the inner attack objectives, we use a momentum factor of 0.9, 200 iterations, and 5 random restarts. The base learning rate (*i.e.*, the attack step size) is varied over the set $\{0.1, 0.01, 0.005\}$ for experiments on MNIST, and over the set $\{0.01, 0.005, 0.001\}$ for experiments on SVHN and CIFAR-10. We report the worst-case results: for each clean input $\mathbf{x}$, if the PGD method with a particular base learning rate can find an $\mathbf{x}'$ that is rejected, then we will use this $\mathbf{x}'$ as the generated adversarial example and consider the inner attack to be successful.

When solving the outer attack objectives, we use a momentum factor of 0.9, 200 iterations, and 5 random restarts. The base learning rate (*i.e.*, the attack step size) is varied over the set $\{0.1, 0.01, 0.005\}$ for experiments on MNIST, and over the set $\{0.01, 0.005, 0.001\}$ for experiments on SVHN and CIFAR-10. We report the worst-case results: for each clean input $\mathbf{x}$, if the PGD method with a particular base learning rate can find an $\mathbf{x}''$ that is accepted and misclassified, then we will use this $\mathbf{x}''$ as the generated adversarial example and consider the outer attack to be successful.

BPDA approximation. Three of the adaptive attacks for CPR (namely CLCIA, PDIA, and CHCMOA) depend on $T(\mathbf{x})$, which does not have a closed-form expression and is not differentiable. Therefore, it is not possible to calculate the exact

gradient for these attack objectives. We will use the BPDA approach (Athalye et al., 2018) to address this challenge, specifically using the straight-through estimator for the gradient.

Recall that $T(\mathbf{x}) \approx \operatorname{argmax}_{\widetilde{\mathbf{x}} \in \mathcal{N}(\mathbf{x}, \widetilde{\epsilon})} \ell_{CE}(\widetilde{\mathbf{x}}, \widehat{y}(\mathbf{x}))$, and it is computed using Algorithm 1. For any of these attack objectives, let $\ell(T(\mathbf{x}))$ denote the terms dependent on $T(\cdot)$. For instance, $\ell(T(\mathbf{x})) = \log h_j(T(\mathbf{x})\,; \boldsymbol{\theta})$ for the CHCMOA attack. Using the chain rule, we can express the gradient of ℓ as follows:

$$\nabla_{\mathbf{x}} \ell(T(\mathbf{x})) \;=\; \mathbf{J}_T(\mathbf{x})^t \, \nabla_{\mathbf{u}} \ell(\mathbf{u})\big|_{\mathbf{u}=T(\mathbf{x})} \tag{42}$$

where $\mathbf{J}_T(\mathbf{x})^t$ is the transpose of the $d \times d$ Jacobian of $T(\mathbf{x})$. For a small $\widetilde{\epsilon}$, we make the approximation that $T(\mathbf{x}) \approx \mathbf{x}$ during the backward pass, which in turn makes $\mathbf{J}_T(\mathbf{x})$ approximately equal to the identity matrix. This gives the following gradient estimate of the BPDA method, which we apply to solve the attack objectives of CLCIA, PDIA, and CHCMOA:

$$\nabla_{\mathbf{x}} \ell(T(\mathbf{x})) \;=\; \nabla_{\mathbf{u}} \ell(\mathbf{u})\big|_{\mathbf{u}=T(\mathbf{x})} \tag{43}$$

During the forward pass, we perform an exact computation of $T(x)$ (using Algorithm 1), but during the backward pass, we approximate the gradient of the attack objectives using Eqn.(43). We note that these adaptive attacks are more expensive to run because they require the computation of $T(\mathbf{x})$ during each PGD step.

E.3. Attack Ensemble

As discussed earlier, for each defense method, we consider an ensemble of inner and outer attacks and report the worst-case robustness with rejection under these attacks. We list the specific attacks in the ensemble for each defense method below:

Confidence-based Rejection. The inner-attack ensemble only the includes Low Confidence Inner Attack (LCIA). The outer-attack ensemble includes the AutoAttack (Croce & Hein, 2020) and High Confidence Misclassification Outer Attack (HCMOA).

RCD. The inner-attack ensemble only includes RCDIA. The outer-attack ensemble includes AutoAttack and RCDOA.

ATRR. The inner-attack ensemble only includes ATRRIA. The outer-attack ensemble includes AutoAttack and ATRROA.

CPR (proposed). The inner-attack ensemble includes LCIA, CLCIA, and PDIA. The outer-attack ensemble includes AutoAttack, HCMOA and CHCMOA. By including a number of strong attacks in the ensemble, we have attempted to perform a thorough evaluation of CPR.

E.4. Evaluating Adaptive Attacks for CPR

In Section 5.2, we perform an ablation study to compare the strength of the different adaptive inner and outer attacks for CPR. These results are in Table 2, and here we discuss the choice of metrics for this evaluation. The outer attack only affects $A_\epsilon^{\text{rej}}(f, 0)$, while the inner attack affects $A_\epsilon^{\text{rej}}(f, \alpha)$ for $\alpha > 0$. Therefore, for the outer attacks we only need to compare $A_\epsilon^{\text{rej}}(f, 0)$. For the inner attacks we compare $A_\epsilon^{\text{rej}}(f, 1)$, while fixing the outer attack to be the strongest ensemble outer attack. This corresponds to the right end of the robustness curve, and gives a clear idea of the strength of the inner attack.

F. Additional Experimental Results

F.1. Evaluating Traditional Metrics

We evaluate our method and the baselines on traditional metrics, including accuracy with rejection on clean test inputs, rejection rate on clean test inputs and robust accuracy with detection defined in (Tramèr, 2022).

The accuracy with rejection on clean test inputs is defined as: $\Pr_{(\mathbf{x},y)\sim\mathcal{D}}\{f(\mathbf{x}) = y \,|\, f(\mathbf{x}) \neq \perp\}$.

The rejection rate on clean test inputs is defined as: $\Pr_{(\mathbf{x},y)\sim\mathcal{D}}\{f(\mathbf{x}) = \perp\}$.

The robust accuracy with detection is defined as:

$$A_\epsilon^{\text{rej}}(f) \;:= 1 - R_\epsilon^{\text{rej}}(f) \;=\; 1 \;-\; \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \left[\mathbf{1}\{f(\mathbf{x}) \neq y\} \vee \max_{\mathbf{x}' \in \mathcal{N}(\mathbf{x},\epsilon)} \mathbf{1}\{f(\mathbf{x}') \notin \{y, \perp\}\} \right].$$

In order to define a single metric that combines the accuracy with rejection and the rejection rate metrics on clean test inputs, we propose to use an F1 score like metric that is based on their harmonic mean:

$$\frac{2\ \Pr_{(\mathbf{x},y)\sim\mathcal{D}}\{f(\mathbf{x}) = y \,|\, f(\mathbf{x}) \neq \perp\}\ \Pr_{(\mathbf{x},y)\sim\mathcal{D}}\{f(\mathbf{x}) \neq \perp\}}{\Pr_{(\mathbf{x},y)\sim\mathcal{D}}\{f(\mathbf{x}) = y \,|\, f(\mathbf{x}) \neq \perp\} + \Pr_{(\mathbf{x},y)\sim\mathcal{D}}\{f(\mathbf{x}) \neq \perp\}}$$

The results for these metrics are given in Table 3. The results show that our method CPR has comparable performance to the baselines on clean test inputs, and also significantly outperforms the baselines on the robust accuracy with detection.

Dataset	Method	Clean Test Inputs			Under Seen Attacks	Under Unseen Attacks
		Acc. with Rej. ↑	Rej. Rate ↓	F1 Score ↑	Robust Acc. with Det. ↑	Robust Acc. with Det. ↑
MNIST	AT	98.81	0.00	99.40	84.70	0.00
	AT+CR	99.55	1.79	98.87	91.60	0.00
	TRADES	99.07	0.00	**99.53**	89.30	0.00
	TRADES+CR	99.67	1.86	98.90	94.00	0.00
	CCAT	99.90	1.82	99.03	83.20	75.50
	RCD	99.02	0.00	99.51	86.50	0.00
	ATRR	99.62	2.51	98.54	91.20	0.00
	AT+CPR (Ours)	99.60	1.99	98.80	**96.10**	**90.40**
	TRADES+CPR (Ours)	99.63	1.63	98.99	95.80	86.70
SVHN	AT	92.58	0.00	96.15	45.10	11.70
	AT+CR	96.22	8.91	93.58	46.10	11.80
	TRADES	92.19	0.00	95.94	52.00	12.30
	TRADES+CR	95.47	9.06	93.15	52.90	12.60
	CCAT	99.04	7.73	95.53	45.30	5.50
	RCD	96.58	0.00	**98.26**	33.80	9.70
	ATRR	96.14	8.98	93.51	44.80	11.50
	AT+CPR (Ours)	95.86	7.34	94.23	55.80	14.70
	TRADES+CPR (Ours)	94.96	6.56	94.20	**62.00**	**18.70**
CIFAR-10	AT	84.84	0.00	91.80	47.60	10.80
	AT+CR	90.55	13.00	88.74	50.00	10.50
	TRADES	82.12	0.00	90.18	48.70	15.20
	TRADES+CR	86.57	9.59	88.45	50.00	15.10
	CCAT	93.18	9.12	92.01	27.70	8.80
	RCD	88.13	2.07	**92.77**	46.70	9.50
	ATRR	89.36	12.09	88.63	48.80	11.30
	AT+CPR (Ours)	89.05	9.57	89.74	56.70	17.10
	TRADES+CPR (Ours)	86.30	9.57	88.32	**57.10**	**21.90**

Table 3. Evaluation of traditional metrics (percentages). Top-1 **boldfaced**.

F.2. Evaluating Robustness Curve

In Section 5.2, we discussed the results of evaluating the robustness curve on the CIFAR-10 dataset. We present the complete results of evaluating the robustness curve on all the datasets under both seen attacks and unseen attacks in Figure 3. The observations on MNIST and SVHN are similar to that of CIFAR-10, except that CCAT has much better robustness with rejection at $\alpha = 0$ than the other baselines on MNIST under unseen attacks.

F.3. Evaluating Total Robust Loss for More Rejection Loss Functions

In this section, we evaluate the total robust loss of the different methods for some additional rejection loss functions. Specifically, we additionally consider the step rejection loss function with $\alpha_0 \in \{0.01, 0.15, 0.2\}$ and the ramp rejection loss function with $t \in \{1, 3\}$. These results are given in Table 4. Our findings are the same as those discussed in Section 5.2.

F.4. Comparing AT with Rejection and AT without Rejection

We compare Adversarial Training (AT) with Consistent Prediction-based Rejection (CPR) to AT without rejection. Both of them use the same base model, which is trained by either standard Adversarial Training (AT) or TRADES. Since adversarial training without rejection accepts every input, the robust accuracy with rejection $A_\epsilon^{\text{rej}}(f, \alpha)$ is a constant (across α), which is

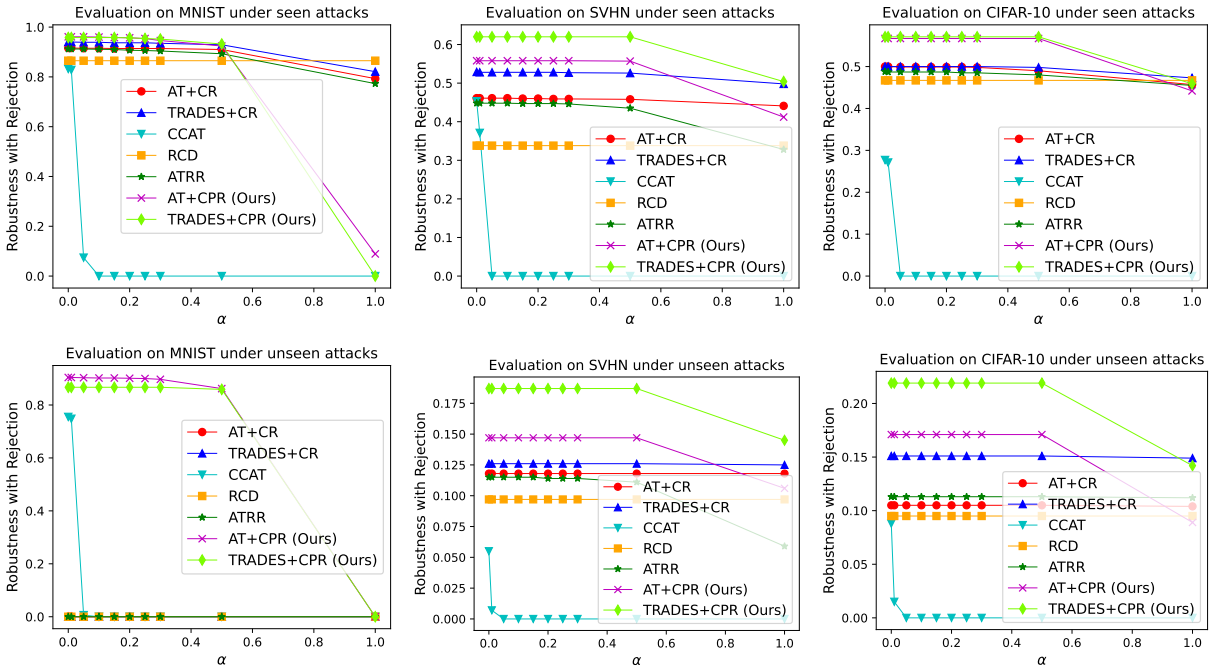

Figure 3. The robustness curve of our method CPR and the baselines.

equal to the standard adversarial robustness. We use AutoAttack (Croce & Hein, 2020) to evaluate the adversarial robustness of adversarial training without rejection.

From the results in Figure 4, we observe that AT with CPR is usually better (has higher robustness with rejection) than AT without rejection, especially under unseen attacks. Under seen attacks, AT without rejection is better than AT with CPR for large α since AT with CPR may reject large perturbations, which is considered to be an error by the robustness with rejection metric for large α. However, if we allow rejecting large perturbations, AT with CPR is always better than AT without rejection, which suggests that including CPR can help improve the performance of adversarial training.

F.5. Ablation Study for CPR

We perform experiments to study the effect of the hyper-parameters $\tilde{\epsilon}$ and m on the proposed method CPR. The results are shown in Figure 5 and Figure 6. From the results, we can see that larger $\tilde{\epsilon}$ leads to better robustness with rejection at $\alpha = 0$. However, it also leads to lower robustness with rejection when α is large, which means CPR rejects more perturbed inputs. Lager $\tilde{\epsilon}$ will also lead to a larger rejection rate on the clean inputs. Besides, larger m also leads to better robustness with rejection at $\alpha = 0$, but may lead to lower robustness with rejection when α is large. Note that larger m leads to more computational cost. We don't need to choose very large m since the results show that as we increase m, the robustness with rejection at $\alpha = 0$ improvement becomes minor.

F.6. Hyper-parameter Selection for CPR

The proposed method CPR has three hyper-parameters: the perturbation budget $\tilde{\epsilon}$, the number of PGD steps m, and the PGD step size η. We don't tune η but just set it to be a fixed value. From Appendix F.5, we know that larger m will lead to better robustness with rejection at $\alpha = 0$. However, as we increase m, the improvement becomes minor. Thus, by considering the computational cost, we select a reasonably large m based on the performance on the validation data. We also know that larger $\tilde{\epsilon}$ will lead to better robustness with rejection at $\alpha = 0$, but will lead to a larger rejection rate on the clean test inputs. In practice, we select a large enough $\tilde{\epsilon}$ such that CPR achieves good robustness with rejection while having a reasonably low rejection rate on the clean inputs. We observed that a wide range of m and $\tilde{\epsilon}$ can lead to good results. Thus, it is easy to select m and $\tilde{\epsilon}$. In our experiments, we consider m in the set $\{10, 20, 30\}$ on MNIST, and m in the set $\{5, 10, 15\}$ on SVHN and CIFAR-10. We consider $\tilde{\epsilon}$ in the set $\{0.05, 0.1, 0.15, 0.2\}$ on MNIST, and $\tilde{\epsilon}$ in the set $\{0.004, 0.005, 0.0055, 0.006\}$ on

Dataset	Method	Total Robust Loss under Seen Attacks ↓					Total Robust Loss under Unseen Attacks ↓				
		Step Rej. Loss			Ramp Rej. Loss		Step Rej. Loss			Ramp Rej. Loss	
		$\alpha_0 = 0.01$	$\alpha_0 = 0.15$	$\alpha_0 = 0.2$	$t = 1$	$t = 3$	$\alpha_0 = 0.01$	$\alpha_0 = 0.15$	$\alpha_0 = 0.2$	$t = 1$	$t = 3$
MNIST	AT+CR	0.084	0.086	0.086	0.117	0.090	1.000	1.000	1.000	1.000	1.000
	TRADES+CR	0.060	0.063	0.063	**0.095**	**0.068**	1.000	1.000	1.000	1.000	1.000
	CCAT	0.172	1.000	1.000	0.972	0.919	0.252	1.000	1.000	0.977	0.934
	RCD	0.135	0.135	0.135	0.135	0.135	1.000	1.000	1.000	1.000	1.000
	ATRR	0.088	0.090	0.093	0.131	0.099	1.000	1.000	1.000	1.000	1.000
	AT+CPR (Ours)	**0.039**	**0.042**	**0.043**	0.295	0.080	**0.096**	**0.098**	**0.099**	**0.340**	**0.136**
	TRADES+CPR (Ours)	0.042	0.042	0.043	0.292	0.078	0.133	0.133	0.133	0.353	0.162
SVHN	AT+CR	0.539	0.540	0.540	0.545	0.541	0.882	0.882	0.882	0.882	0.882
	TRADES+CR	0.472	0.472	0.472	0.480	0.473	0.874	0.874	0.874	0.874	0.874
	CCAT	0.629	1.000	1.000	0.988	0.967	0.993	1.000	1.000	1.000	0.999
	RCD	0.662	0.662	0.662	0.662	0.662	0.903	0.903	0.903	0.903	0.903
	ATRR	0.552	0.553	0.553	0.587	0.559	0.885	0.885	0.886	0.901	0.888
	AT+CPR (Ours)	0.442	0.442	0.442	0.479	0.447	0.853	0.853	0.853	0.863	0.854
	TRADES+CPR (Ours)	**0.380**	**0.380**	**0.380**	**0.409**	**0.384**	**0.813**	**0.813**	**0.813**	**0.823**	**0.814**
CIFAR-10	AT+CR	0.500	0.501	0.501	0.515	0.504	0.895	0.895	0.895	0.895	0.895
	TRADES+CR	0.500	0.500	0.500	0.507	0.501	0.849	0.849	0.849	0.849	0.849
	CCAT	0.729	1.000	1.000	0.992	0.976	0.985	1.000	1.000	0.999	0.998
	RCD	0.533	0.533	0.533	0.533	0.533	0.905	0.905	0.905	0.905	0.905
	ATRR	0.513	0.513	0.513	0.524	0.516	0.887	0.887	0.887	0.887	0.887
	AT+CPR (Ours)	0.433	0.433	0.433	0.464	0.437	0.829	0.829	0.829	0.849	0.832
	TRADES+CPR (Ours)	**0.429**	**0.429**	**0.429**	**0.457**	**0.433**	**0.781**	**0.781**	**0.781**	**0.800**	**0.783**

Table 4. The total robust losses for different rejection loss functions. The best result is **boldfaced**.

SVHN and CIFAR-10. We select the best m and $\tilde{\epsilon}$ based on the performance on the validation data.

F.7. Attacking CCAT

In this section, we show that the High Confidence Misclassification Outer Attack (HCMOA) with attack step size enumeration proposed in Appendix E is stronger than the PGD attack with backtracking proposed in (Stutz et al., 2020) for evaluating robustness with detection of CCAT. We follow the setup in (Stutz et al., 2020) to train CCAT models and attack CCAT. For the PGD attack with backtracking, we use a base learning rate of 0.001, momentum factor of 0.9, learning rate factor of 1.1, 1,000 iterations, and 10 random restarts.

Dataset	Attack	Robustness with Detection ↓	
		Seen Attacks	Unseen Attacks
MNIST	PGD with backtracking	88.50	85.30
	HCMOA	**83.20**	**75.50**
SVHN	PGD with backtracking	64.10	62.30
	HCMOA	**45.30**	**9.10**
CIFAR-10	PGD with backtracking	43.70	38.20
	HCMOA	**27.70**	**8.80**

Table 5. Attacking CCAT using different attacks. All numbers are percentages. **Bold** numbers are superior results.

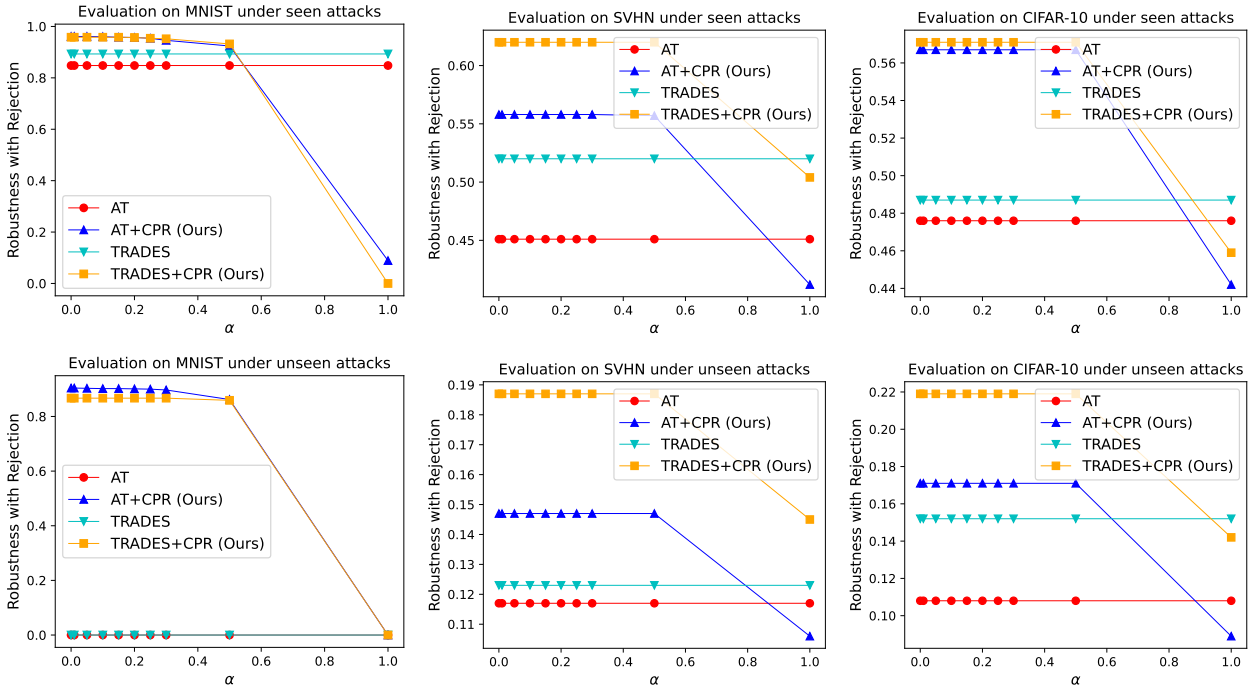

Figure 4. Results of comparing the robustness with rejection of AT with CPR and AT without CPR.

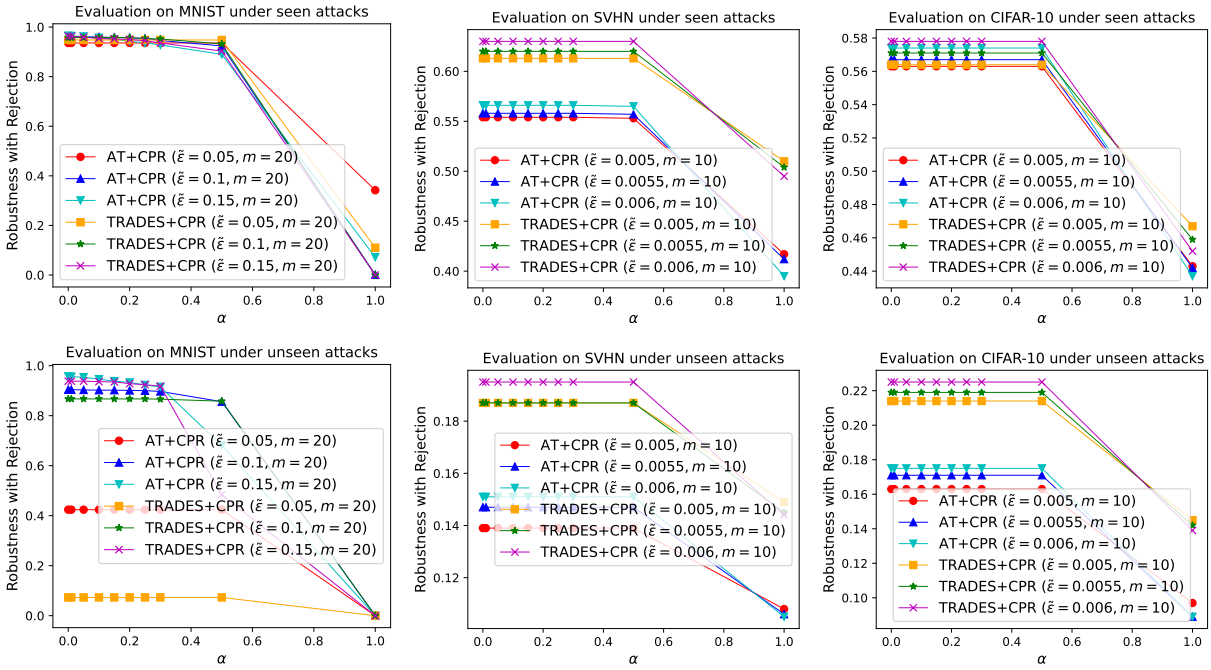

Figure 5. Ablation study for the proposed method CPR where we vary the hyper-parameter $\tilde{\epsilon}$ while fixing the hyper-parameter m.

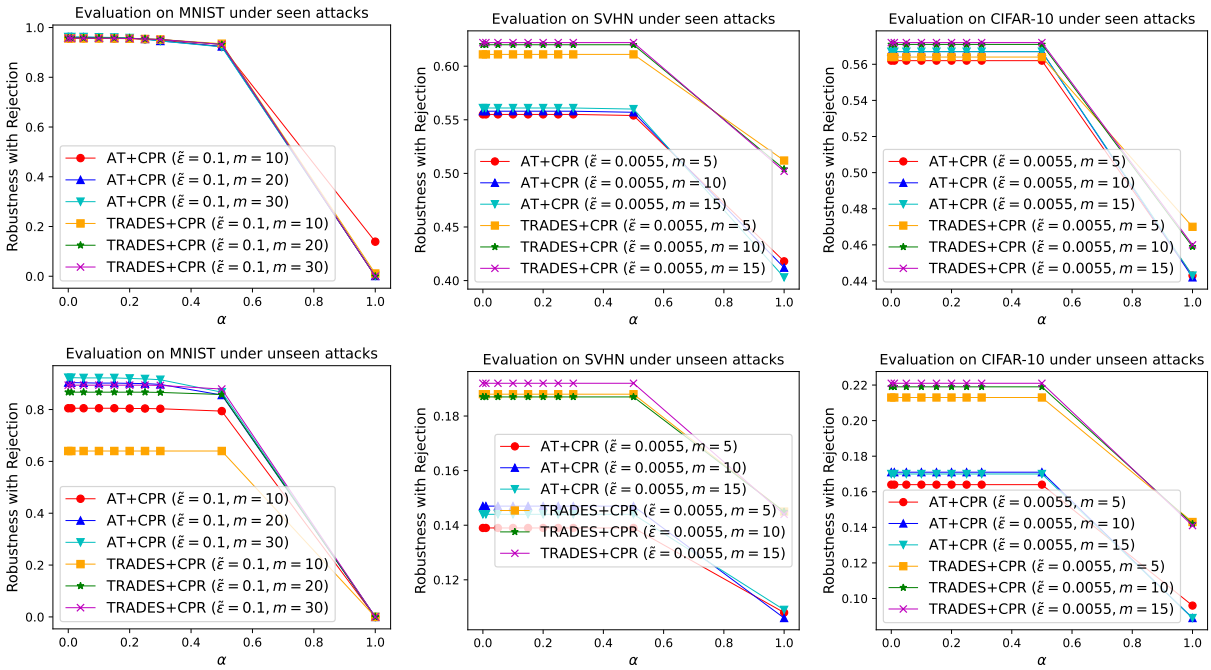

Figure 6. Ablation study for the proposed method CPR where we vary the hyper-parameter m while fixing the hyper-parameter $\tilde{\varepsilon}$.