Deep neural networks (DNNs) are vulnerable to adversarial examples crafted by imperceptible perturbations. A range of defense techniques have been proposed to improve DNN robustness to adversarial examples, among which adversarial training has been demonstrated to be the most effective. Adversarial training is often formulated as a min-max optimization problem, with the inner maximization for generating adversarial examples. However, there exists a simple, yet easily overlooked fact that adversarial examples are only defined on correctly classified (natural) examples, but inevitably, some (natural) examples will be misclassified during training. In this paper, we investigate the distinctive influence of misclassified and correctly classified examples on the final robustness of adversarial training. Specifically, we find that misclassified examples indeed have a significant impact on the final robustness. More surprisingly, we find that different maximization techniques on misclassified examples may have a negligible influence on the final robustness, while different minimization techniques are crucial. Motivated by the above discovery, we propose a new defense algorithm called Misclassification Aware adveRsarial Training (MART), which explicitly differentiates the misclassified and correctly classified examples during the training. We also propose a semi-supervised extension of MART, which can leverage the unlabeled data to further improve the robustness. Experimental results show that MART and its variant could significantly improve the state-of-the-art adversarial robustness.

1 INTRODUCTION

Despite their great success in applications such as computer vision (He et al., 2016), speech recognition (Wang et al., 2017) and natural language processing (Devlin et al., 2018), deep neural networks (DNNs) are extremely vulnerable to adversarial examples crafted by adding small adversarial perturbations to natural examples (Szegedy et al., 2013; Goodfellow et al., 2015). Given a DNN classifier $h_\theta$ with parameter $\theta$ and a correctly classified natural example $x$ with class label $y (h_\theta(x) = y)$, an adversarial example $x'$ can be generated by perturbing $x$ such that $h_\theta(x') \neq y$, i.e., the natural example is correctly classified before perturbation but misclassified after perturbation. The perturbation required for misclassification is often small and bounded by an $L_p$-norm $\|x' - x\|_p \leq \epsilon$, which keeps $x'$ within the $\epsilon$-ball centered at $x$, so that it is visually the “same” for human observers. This vulnerability of DNNs raises serious security concerns about their practicability in security critical applications, e.g., face recognition (Kurakin et al., 2016) and autonomous driving (Chen et al., 2015).

Compared with pre/post-processing methods such as feature squeezing (Xu et al., 2017), input denoising (Guo et al., 2018; Liao et al., 2018; Samangouei et al., 2018) and adversarial detection (Feinman et al., 2017; Grosse et al., 2017; Meng & Chen, 2017; Ma et al., 2018), several defense techniques have been proposed to train DNNs that are inherently robust to adversarial examples including defensive distillation (Papernot et al., 2016), gradient regularization (Gu & Rigazio, 2014; Papernot et al., 2017), Ross & Doshi-Velez (2018; Tramer et al., 2018), model compression (Das et al., 2018; Liu et al., 2018), and activation pruning (Bhillion et al., 2018; Rakin et al., 2018), among which adversarial training has been demonstrated to be the most effective (Goodfellow et al., 2015; Athalye et al., 2018; Madry et al., 2018). Adversarial training can be regarded as a data augmentation technique that trains DNNs on adversarial examples, and can be viewed as solving the following
In this paper, both “correctly classified” and “misclassified” refer to predictions on natural training examples.
Under review as a conference paper at ICLR 2020

and step size $\epsilon/4$, then use this network (87% training accuracy) to select two subsets of natural training examples to investigate: 1) a subset of misclassified examples $S^-$ (13% of training data), and 2) a subset of correctly classified examples $S^+$ (also 13% of training data, $|S^+| = |S^-|$). Using these two subsets, we explore different ways to re-train the same network, and evaluate its robustness against white-box PGD\textsuperscript{20} (step size $\epsilon/10$) attacks on the test dataset.

In Figure 1(a) we find that misclassified examples have a significant impact on the final robustness. Compared with standard adversarial training (dashed blue line), the final robustness drops drastically, if examples in subset $S^-$ are not perturbed (solid green line) during adversarial training (other examples are still perturbed by PGD\textsuperscript{10}). In contrast, the same operation on subset $S^+$ only slightly affects the final robustness (solid orange line). Previous work has found that removing a small proportion of training examples does not reduce the robustness (Ding et al., 2019), which seems to be true for correctly classified examples, but is apparently not true for misclassified examples.

To further understand the distinctive influence of misclassified and correctly classified examples, we test different techniques on them within either the maximization or the minimization process of adversarial training. Firstly, we apply different maximization techniques while keeping the minimization loss CE unchanged. As shown in Figure 1(b) the final robustness is barely affected when we use a weak attack (e.g., Fast Gradient Sign Method (FGSM) (Goodfellow et al., 2015)) to perturb misclassified examples $S^-$ (all other training examples are still perturbed by PGD\textsuperscript{10}). This suggests that different maximization techniques on misclassified examples $S^-$ may have a negligible influence on the final robustness, provided that the inner maximization problem is solved to a moderate precision (Wang et al., 2019). However, for subset $S^+$, a weak attack for the maximization tends to degenerate the robustness. Secondly, we test different minimization techniques with the inner maximization still solved by PGD\textsuperscript{10}. Interestingly, we find that different minimization techniques on misclassified examples make a significant difference to the final robustness. As shown in Figure 1(c) compared with standard adversarial training (dashed blue line) with the CE loss, the final robustness is significantly improved when the outer minimization on misclassified examples is “regularized” (solid green line) by an additional term (a KL-divergence term that was used previously in Zheng et al. (2016), Zhang et al. (2019)). The same regularization applied to correctly classified examples also helps the final robustness (solid orange line), though not as significantly as for misclassified examples.

Motivated by the above observations, we reformulate the adversarial risk to incorporate an explicit differentiation of misclassified examples in a form of regularization. We then propose a new defense algorithm to achieve this in a dynamic way during adversarial training. Our main contributions are:

- We investigate the distinctive influence of misclassified and correctly classified examples on the final robustness of adversarial training. We find that the manipulation on misclassified examples has more impact on the final robustness, and the minimization techniques are more crucial than maximization ones under the min-max optimization framework.

- We propose a regularized adversarial risk which incorporates an explicit differentiation of misclassified examples as a regularizer. Based on that, we further propose a new defense algorithm, called Misclassification Aware adversarial Training (MART).

- Experimentally, we show that adversarial robustness can be significantly improved over the state-of-the-art, by a specific focus on misclassified examples. It also helps improve recently proposed adversarial training with unlabeled data.

2 Misclassification Aware Adversarial Risk

In this section, we propose a regularized adversarial risk that incorporates an explicit differentiation of misclassified examples.

2.1 Preliminaries

We first define some notations. We use lower case and lower case bold face to denote scalars and vectors, respectively. We use upper case calligraphic symbols to denote sets.

For a $K$-class ($K \geq 2$) classification problem, given a dataset $\{(x_i, y_i)\}_{i=1, \ldots, n}$ with $x_i \in \mathbb{R}^d$ as a natural example and $y_i \in \{1, \ldots, K\}$ as its associated label, a DNN classifier $h_\theta$ with model
parameter $\theta$ predicts the class of an input example $x_i$:

$$h_\theta(x_i) = \arg \max_{k=1,...,K} p_k(x_i, \theta), \quad p_k(x_i, \theta) = \exp(z_k(x_i, \theta))/\sum_{k'=1}^K \exp(z_{k'}(x_i, \theta)), \quad (2)$$

where $z_k(x_i, \theta)$ is the logits output of the network with respect to class $k$, and $p_k(x_i, \theta)$ is the probability (softmax on logits) of $x_i$ belonging to class $k$.

The adversarial risk [Madry et al., 2018] on dataset $\{(x_i, y_i)\}_{i=1,...,n}$ and classifier $h_\theta$ can be defined with respect to the 0-1 loss [Zhang et al., 2019] as:

$$\mathcal{R}(h_\theta) = \frac{1}{n} \sum_{i=1}^n \max_{x' \in B_\epsilon(x_i)} \mathbb{1}(h_\theta(x'_i) \neq y_i), \quad (3)$$

where $\mathbb{1}(\cdot)$ is the indicator function and $B_\epsilon(x_i) = \{x : \|x - x_i\|_p \leq \epsilon\}$ denotes the $L_p$-norm ball centered at $x_i$ with radius $\epsilon$. We will focus on the $L_\infty$-ball in this paper.

### 2.2 Misclassification Aware Regularization

Note that the adversarial risk in (3) is defined on adversarial examples within the $\epsilon$-ball of all natural examples, regardless of whether they are correctly classified ($h_\theta(x_i) = y_i$) or misclassified ($h_\theta(x_i) \neq y_i$) by the current model $h_\theta$. To differentiate, we reformulate the adversarial risk based on the prediction of the current network $h_\theta$. Specifically, natural training examples can be divided into two subsets with respect to $h_\theta$, with one subset of correctly classified examples ($S_{h_\theta}^+$) and one subset of misclassified examples ($S_{h_\theta}^-$):

$$S_{h_\theta}^+ = \{i : i \in [n], h_\theta(x_i) = y_i\} \quad \text{and} \quad S_{h_\theta}^- = \{i : i \in [n], h_\theta(x_i) \neq y_i\}.$$ 

Then we are going to define adversarial risk separately for correctly classified and misclassified examples. As we observed in Figure [1c], regularization on misclassified examples can significantly improve robustness. Therefore, for misclassified examples, we formulate the adversarial risk as:

$$\mathcal{R}^-(h_\theta, x_i) := \mathbb{1}(h_\theta(x'_i) \neq y_i) + \mathbb{1}(h_\theta(x_i) \neq h_\theta(x'_i)), \quad (4)$$

where the adversarial example $x'_i$ is generated by solving

$$\begin{align*}
x'_i = \arg \max_{x' \in B_\epsilon(x_i)} \mathbb{1}(h_\theta(x'_i) \neq y_i). \quad (5)
\end{align*}$$

We remark that the first and second terms on the R.H.S. of (4) correspond to the standard adversarial risk and the regularization term respectively. Moreover, we would like to clarify that the regularization term $\mathbb{1}(h_\theta(x'_i) \neq h_\theta(x'_i))$ aims to encourage the output of neural network to be stable against misclassified adversarial examples. For misclassified examples, direct minimization of the standard adversarial risk may be too hard, as themselves cannot be classified correctly, even without any perturbations. A similar idea has been used in stability training on all training examples [Zheng et al., 2016; Kannan et al., 2018; Zhang et al., 2019].

Then we consider correctly classified examples. As can be observed in Figure [1c], regularization on correctly classified examples cannot provide as significant improvement as achieved by that on misclassified ones. Moreover, in this case it can be found that $\mathbb{1}(h_\theta(x_i) \neq h_\theta(x'_i)) = \mathbb{1}(h_\theta(x'_i) \neq y_i)$ since we have $h_\theta(x_i) = y_i$, which implies that the regularizer has exactly same form as the adversarial risk. Therefore, for correctly classified example, we simply use the standard adversarial risk, i.e.,

$$\mathcal{R}^+(h_\theta, x_i) := \max_{x' \in B_\epsilon(x_i)} \mathbb{1}(h_\theta(x'_i) \neq y_i) = \mathbb{1}(h_\theta(x'_i) \neq y_i), \quad (6)$$

where the adversarial example $x'_i$ is defined in (5).

Finally, combining the proposed two adversarial risks for correctly classified examples and misclassified examples in an adversarial training framework, we can train a network that minimizes the following risk:

$$\min_{\theta} \mathcal{R}_{\text{misc}}(h_\theta) := \frac{1}{n} \left( \sum_{i \in S_{h_\theta}^+} \mathcal{R}^+(h_\theta, x_i) + \sum_{i \in S_{h_\theta}^-} \mathcal{R}^-(h_\theta, x_i) \right)$$

$$= \frac{1}{n} \sum_{i=1}^n \left\{ \mathbb{1}(h_\theta(x_i) = y_i) \cdot R^+(h_\theta, x_i) + \mathbb{1}(h_\theta(x_i) \neq y_i) \cdot R^-(h_\theta, x_i) \right\}$$

$$= \frac{1}{n} \sum_{i=1}^n \left\{ \mathbb{1}(h_\theta(x_i) = y_i) \cdot \mathbb{1}(h_\theta(x'_i) \neq y_i) \right.$$ 

$$+ \mathbb{1}(h_\theta(x_i) \neq y_i) \cdot \left( \mathbb{1}(h_\theta(x'_i) \neq y_i) + \mathbb{1}(h_\theta(x_i) \neq h_\theta(x'_i)) \right) \right\}$$

$$= \frac{1}{n} \sum_{i=1}^n \left\{ \mathbb{1}(h_\theta(x'_i) \neq y_i) + \mathbb{1}(h_\theta(x_i) \neq h_\theta(x'_i)) \cdot \mathbb{1}(h_\theta(x_i) \neq y_i) \right\}, \quad (7)$$
where \( \hat{x}_i' \) is defined in (5) and the second equality follows from the definition of \( S_{h_\theta} \) and \( \hat{S}_{h_\theta} \). The new risk defined above is a regularized adversarial risk with regularization term \( \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}(h_\theta(x_i) \neq h_\theta(\hat{x}_i')) \cdot \mathbb{I}(h_\theta(x_i) \neq y_i) \), which we call the misclassification aware regularization.

3 Proposed Defense: Misclassification Aware Adversarial Training (MART)

In the previous section, we derived the misclassification aware adversarial risk based on 0-1 loss. However, optimization over 0-1 loss is intractable in practice. We next propose a Misclassification Aware Adversarial Training (MART) algorithm, by replacing the 0-1 losses with proper surrogate loss functions which are both physical meaningful and computationally tractable. Following that, we further analyze the difference of MART to existing work, and propose a semi-supervised extension.

3.1 The Proposed Defense Algorithm

Surrogate Loss for Outer Minimization. As presented in (7), the minimization consists of three indicator functions: (1) \( \mathbb{I}(h_\theta(\hat{x}_i') \neq y_i) \); (2) \( \mathbb{I}(\hat{h}_\theta(x_i) \neq h_\theta(x_i')) \); and (3) \( \mathbb{I}(h_\theta(x_i) \neq y_i) \).

For the first indicator function \( \mathbb{I}(h_\theta(\hat{x}_i') \neq y_i) \), we propose to use a boosted cross entropy (BCE) loss as the surrogate loss, instead of the commonly used CE loss in (Madry et al., 2018; Wang et al., 2019). This is largely because classifying adversarial examples requires a stronger classifier than natural examples, as the presence of adversarial examples makes the classification decision boundary become more complicated. This is pointed out by (Madry et al., 2018), where they increase the model capacity for a stronger classifier. The benefit of using BCE compared to CE will shortly be presented in the experiment section. The proposed BCE loss is defined as:

\[
\text{BCE}(p(\hat{x}_i', \theta), y_i) = -\log (p_{y_i}(\hat{x}_i', \theta)) - \log (1 - \max_{k \neq y_i} p_k(\hat{x}_i', \theta)),
\]

(8)

where \( p_k(\hat{x}_i', \theta) \) is the probability output defined in (2), the first term \( -\log (p_{y_i}(\hat{x}_i', \theta)) \) is the commonly used CE loss, denoted CE\((p(\hat{x}_i', \theta), y_i)\), and the second term \( -\log (1 - \max_{k \neq y_i} p_k(\hat{x}_i', \theta)) \) is a margin term used to improve the decision margin of the classifier. A similar idea has been used for improving adversarial strength by Carlini & Wagner (2017). Note that BCE is just a simple boost that works well in our experiments and other boosted losses could also work here.

For the second indicator function \( \mathbb{I}(\hat{h}_\theta(x_i) \neq h_\theta(\hat{x}_i')) \), we can use KL divergence as the surrogate loss function (Zhang et al., 2019; Zheng et al., 2016), since \( h_\theta(x_i) \neq h_\theta(\hat{x}_i') \) implies that adversarial examples have different output distributions to that of natural examples. Thus, we have

\[
\text{KL}(p(x_i, \theta) || p(\hat{x}_i', \theta)) = \sum_{k=1}^{K} p_k(x_i, \theta) \log \frac{p_k(x_i, \theta)}{p_k(\hat{x}_i', \theta)}.
\]

(9)

The third indicator function \( \mathbb{I}(h_\theta(x_i) \neq y_i) \) is a condition that emphasizes learning on misclassified examples. However, the condition cannot be directly optimized if we conduct a hard decision during the training process (Ding et al., 2019) uses hard decision and does not optimize the condition). Instead, we propose to use a soft decision scheme by replacing \( \mathbb{I}(h_\theta(x_i) \neq y_i) \) with the output probability \( 1 - p_{y_i}(x_i, \theta) \). This will be large for misclassified examples and small for correctly classified examples.

Surrogate Loss for Inner Maximization. The goal of inner maximization is to generate adversarial example \( \hat{x}_i' \) for natural example \( x_i \) by solving (5). Therefore, we aim to find a surrogate loss function for the indicator function \( \mathbb{I}(h_\theta(\hat{x}_i') \neq y_i) \). Here, we leverage the commonly used CE loss as the surrogate loss and find the adversarial example \( \hat{x}_i' \) as follows:

\[
\hat{x}_i' = \arg \max_{x_i' \in S_\theta}(x_i) \text{CE}(p(x_i', \theta), y_i).
\]

(10)

Following our findings in Figure (1(b)) that a strong attack can help robustness (though is negligible for misclassified examples), we propose to use the (strong) PGD attack to maximize the CE loss for both the correctly classified and misclassified examples, the same as standard adversarial training.\footnote{We slightly abuse the notation of \( \hat{x}_i' \) as the maximizer of the surrogate loss rather than the 0-1 loss in (5).}
Note that other surrogate loss functions, such as those exploited in (Athalye et al., 2018; Carlini & Wagner, 2017) for adversarial attack, could also be used here.

The Overall Objective. Based on the surrogate loss functions, we can state the final objective function for our proposed Misclassification Aware adveRsarial Training (MART) defense:

$$
\mathcal{L}_{\text{MART}}(\theta) = \frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i, \theta),
$$

where $\ell(x_i, y_i, \theta)$ is defined as

$$
\ell(x_i, y_i, \theta) := \text{BCE}(p(\hat{x}_i', \theta), y_i) + \lambda \cdot \text{KL}(p(x_i, \theta)\|p(\hat{x}_i', \theta)) \cdot (1 - p_{y_i}(x_i, \theta)).
$$

Here the adversarial example $\hat{x}_i'$ is generated by (10), and $\lambda$ is a tunable scaling parameter that balances the two parts of the final loss, and is fixed for all training examples. The complete training procedure of MART is described in Appendix A.

3.2 Relation to Existing Work

In this section, we briefly discuss the difference between our MART and existing defense methods including standard adversarial training (Standard) (Madry et al., 2018), logit pairing methods (Kannan et al., 2018), max-margin adversarial training (MMA) (Ding et al., 2018) and TRADES (Zhang et al., 2019), as presented in Table 1.

Table 1: Loss function comparison with existing work. The adversarial example $\hat{x}_i'$ is generated by (10) for all defense methods except TRADES and MMA. The adversarial example in TRADES is generated by maximizing its regularization term (KL-divergence), and the adversarial example in MMA is generated by solving (10) with different perturbation limit ($i.e., \epsilon$).

<table>
<thead>
<tr>
<th>Defense Method</th>
<th>Loss Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>CE(p(\hat{x}', \theta), y)</td>
</tr>
<tr>
<td>ALP</td>
<td>CE(p(\hat{x}', \theta), y) + \lambda \cdot</td>
</tr>
<tr>
<td>CLP</td>
<td>CE(p(x, \theta), y) + \lambda \cdot</td>
</tr>
<tr>
<td>TRADES</td>
<td>CE(p(x, \theta), y) + \lambda \cdot \text{KL}(p(x, \theta)|p(\hat{x}', \theta))</td>
</tr>
<tr>
<td>MMA</td>
<td>CE(p(\hat{x}', \theta), y) \cdot 1(h_\theta(x) = y) + CE(p(x, \theta), y) \cdot 1(h_\theta(x) \neq y)</td>
</tr>
<tr>
<td>MART</td>
<td>BCE(p(\hat{x}', \theta), y) + \lambda \cdot \text{KL}(p(x, \theta)|p(\hat{x}', \theta)) \cdot (1 - p_{y_i}(x, \theta))</td>
</tr>
</tbody>
</table>

Specifically, the Standard algorithm was designed to minimize the standard adversarial loss, $i.e.,$ cross-entropy loss on adversarial examples. Logit pairing methods, consisting of adversarial logit pairing (ALP) and clean logit pairing (CLP), introduce a regularization term enclosing both natural examples and their adversarial counterparts. The objective function of TRADES is also a linear combination of natural loss and regularization terms on the output probabilities corresponding to natural examples and their adversarial counterparts using KL divergence. However, none of these algorithms differentiates the misclassified examples and correctly classified examples.

The most relevant work is MMA, which proposes to use maximal margin optimization for correctly classified examples while keeping the optimization on misclassified examples unchanged. Specifically, for correctly classified examples, MMA adopts cross-entropy loss on adversarial examples, which are generated by solving (10) with example-dependent perturbation limit. For misclassified examples, MMA directly applies cross-entropy loss on natural examples. We emphasize that our MART is different from MMA in the following aspects: (1) MMA performs hard decision to identify misclassified examples from training data, while MART uses soft decision scheme on training data based on the corresponding output probabilities (p(\hat{x}', \theta)), which can be jointly learned during the training process; (2) for correctly classified examples, MMA adopts cross-entropy loss on adversarial examples with different perturbation limits, while MART utilizes the proposed BCE loss on the adversarial examples with the same perturbation limit; (3) for misclassified examples, MMA adopts cross-entropy loss on natural examples, while MART adopts a regularized adversarial loss involving both adversarial and natural examples. Because of these differences, we later will show that MART outperforms MMA in the experiments.

3.3 Semi-supervised Extension with Unlabeled Data

Recent work has shown that semi-supervised learning with additional unlabeled data can improve the adversarial robustness (Uesato et al., 2019; Carmon et al., 2019; Najafi et al., 2019; Zhai et al., 2019).
Specifically, the training loss function applied in these semi-supervised learning methods is typically defined as a weighted sum of the supervised loss (loss on the labeled data) and the unsupervised one (loss on the unlabeled data), i.e.,

\[ \mathcal{L}(\theta) = \mathcal{L}_{\text{sup}}(\theta) + \gamma \cdot \mathcal{L}_{\text{unsup}}(\theta), \]

where \( \gamma > 0 \) is the weight of unsupervised loss. As pointed out in Uesato et al. (2019), there are multiple choices of the unsupervised loss function \( \mathcal{L}_{\text{unsup}}(\theta) \), leading to different defense methods, among which the most effective defense method is UAT++. In particular, UAT++ first trains a natural model on labeled data, and then use this model to generate pseudo labels for unlabeled data. Moreover, given a training data \((x, y)\) (which can be either labeled or unlabeled), the supervised and unsupervised loss functions adopted in UAT++ are defined as

\[ \ell^{\text{UAT++}}_{\text{sup}}(x, y; \theta) = \max_{x' \in B_x} \mathcal{L}_{\text{sup}}(x, y) = \max_{x' \in B_x} \mathcal{CE}(p(x', \theta), y) + \lambda \cdot \max_{x' \in B_x} \mathcal{KL}(p(x, \theta)||p(x', \theta)), \]

where \( \lambda \) is a tunable hyperparameter. A similar idea was also proposed in a concurrent work (Carmon et al., 2019), leading to another semi-supervised defense method called RST. The first stage of RST is also to generate pseudo labels for unlabeled data by training a natural model on the labeled data. Then in the second stage, RST applies TRADES loss to train the robust model based on both labeled and unlabeled data, i.e., given a training data \((x, y)\), the supervised and unsupervised loss functions adopted in RST are defined as

\[ \ell^{\text{RST}}_{\text{sup}}(x, y; \theta) = \mathcal{L}_{\text{sup}}(x, y) = \mathcal{CE}(p(x, \theta), y) + \lambda \cdot \mathcal{KL}(p(x, \theta)||p(x', \theta)). \]

As we pointed out in Figure 1(b) and the following experiment section, the maximization technique has a neglectable influence on the robustness. Therefore, the major difference between UAT++ and RST is the objective function for minimization. Considering MART is also an objective function, it thus could be easily combined with semi-supervised learning with unlabeled data. Following RST, we propose the following semi-supervised version of MART:

\[ \mathcal{L}_{\text{semi}}^{\text{MART}}(\theta) = \sum_{i \in S_{\text{sup}}} \ell^{\text{MART}}_{\text{sup}}(x_i, y_i; \theta) + \gamma \cdot \sum_{i \in S_{\text{unsup}}} \ell^{\text{MART}}_{\text{unsup}}(x_i, y_i; \theta) \]

with supervised and unsupervised loss function defined as follows,

\[ \ell^{\text{MART}}_{\text{sup}}(x, y; \theta) = \max_{x'} \mathcal{MART}(x, y; \theta) = \mathcal{BCE}(p(x', \theta), y) + \lambda \cdot \mathcal{KL}(p(x, \theta)||p(x', \theta)) \cdot (1 - p(x, \theta)), \]

where the adversarial example \( \hat{x}' \) is generated by solving (10), and \( S_{\text{sup}} \) and \( S_{\text{unsup}} \) denote the set of labeled data and unlabeled data respectively.

## 4 Experiments

In this section, we first conduct a set of experiments to provide a comprehensive understanding of our proposed defense MART, and then evaluate its robustness on benchmark datasets in both white-box and black-box settings. Finally, we benchmark the state-of-the-art robustness and explore using unlabeled data for further improvement.

### 4.1 Understanding the Proposed MART

Here, we investigate MART from 4 different perspectives: (1) removing components of the MART loss function, (2) replacing components of the MART loss function, (3) misclassification aware loss on certain proportions of training data, and (4) sensitivity to regularization parameter \( \lambda \).

**Experimental Setup.** We train ResNet-18 (He et al., 2016) with different variants of MART on CIFAR-10 (Krizhevsky & Hinton, 2009). All the models are trained using SGD with momentum 0.9, weight decay \( 2 \times 10^{-4} \) and an initial learning rate of 0.1, which is divided by 10 at the 75-th and 90-th epoch (100 epochs in total). All natural images are normalized into \([0, 1]\), and simple data augmentations including 4-pixel padding with \(32 \times 32\) random crop and random horizontal flip. The maximum perturbation \( \epsilon = 8/255 \) and parameter \( \lambda = 6 \). The training attack is PGD\(^{10}\) with random start and step size \( \epsilon/4 \), while the test attack is PGD\(^{20}\) with random start and step size \( \epsilon/10 \).

**Removing Components of MART.** Recalling the objective function of MART in (11), it has three terms in the loss function: BCE, KL and \( 1 - p \)^\(4\). As illustrated in Figure 2(a), removing \( 1 - p \) or KL

\(^1\)The adversarial example \( \hat{x}' \) in KL term is reused the one in CE term for efficiency in Uesato et al. (2019)

\(^4\)For simplicity, we use these abbreviations in Section 4.1
or both all leads to a significant robustness degradation. In particular, we found that the soft decision term \(1 - p\) has a constant robustness improvement throughout the training process, while the KL term can help mitigate overfitting at a later stage of training (after 80 epochs). When the two terms are combined together, they boost the final robustness considerably without causing overfitting.

**Replacing Components of MART.** As we show in Figure 2(b), when the BCE component is either replaced by a CE term or redefined on natural examples \((x_{\text{nat}})\), the final robustness decreases by a substantial amount. It suggests that learning with CE instead of our proposed BCE suffers from insufficient learning with lower robustness throughout the entire training process. On the other hand, learning with BCE on natural examples exhibits severe overfitting at the later stage (solid green line). We did not observe any benefit when replacing CE by KL in the inner maximization of adversarial min-max framework (solid red line), an observation that is consistent with Figure 2(b).

**Ablation on Training Data.** Here, we show the contribution of our proposed misclassification aware regularization \((e.g., \text{KL} \cdot (1 - p))\) to the final robustness with respect to the training data. Specifically, we gradually increase the proportion of training examples that are trained using the proposed misclassification aware regularization term, and display the corresponding robustness in Figure 2(c). The training examples using the proposed regularization are randomly selected, and the BCE term is still defined on all training (adversarial) examples. As can be observed, the robustness can be improved steadily when the proposed regularization is applied on more data. This verifies the benefit of the differentiation of correctly classified and misclassified examples.

**Sensitivity to Regularization Parameter \(\lambda\).** We further investigate the parameter \(\lambda\) in MART objective function defined in (11) which controls the strength of the regularization. We also test the regularization parameter \(\lambda\) of TRADES (please refer to Table 1). We present the results in Figure 2(d) for different \(\lambda \in \{1/2, 50\}\). By explicitly differentiating the misclassified and correctly classified examples, MART achieves good stability and robustness across different choices of \(\lambda\), which is also consistently better and more stable than that of TRADES.

### 4.2 Robustness Evaluation and Analysis

In this part, we evaluate the robustness of MART on both MNIST (LeCun et al., 1998) and CIFAR-10 (Krizhevsky & Hinton, 2009) datasets against various white-box and black-box attacks.

**Baselines.** (1) Standard (Madry et al., 2018); (2) MMA (Ding et al., 2019); (3) Dynamic (Wang et al., 2019); and (4) TRADES (Zhang et al., 2019). We only compare with adversarial training variants, since they are the most effective defense to date (Athalye et al., 2018).

**Defense Settings.** For MNIST, all defense models are built on a 4-layer CNN and trained using SGD with momentum 0.9. The initial learning rate is 0.01 and divided by 10 at the 20-th and 40-th epoch (50 epochs in total). For CIFAR-10, the settings are the same as Section 4.1. For the training attack, it is also the PGD with random start and step size \(\epsilon/4\). The perturbation limit \(\epsilon = 0.3\) for MNIST, and \(\epsilon = 8/255\) for CIFAR-10. For MART, we set \(\lambda = 6\). Other hyperparameters of the baselines are configured as per their original papers: max margin is set to 0.45 (MNIST) or 12/255 (CIFAR-10) for MMA, maximum criterion value \(c_{\text{max}} = 0.5\) for Dynamic and \(\lambda = 6\) for TRADES.

---

3For clarity, we set the ticks on x-axis in uniform interval for different \(\lambda\).
4Standard, MMA and TRADES have been introduced in Section 3.2 Dynamic is the adversarial training with a criterion that dynamically controls the convergence quality of the inner maximization.
White-box Robustness. We evaluate the robustness of all defense models against three types of attacks for both MNIST and CIFAR-10: FGSM, PGD\(^{20}\) (20-step PGD with step size \(\epsilon/10\)), and CW\(_\infty\) (\(L_\infty\) version of CW optimized by PGD). All attacks have full access to model parameters and are constrained by the same perturbation limit \(\epsilon\). The white-box robustness of all defense models are reported in Table 2, where “Natural” denotes the accuracy on natural test images. Our proposed defense MART achieves the best robustness against all three types of attacks on both MNIST and CIFAR-10. Compared with MNIST, the robustness improvements of MART over other baselines are more significant on CIFAR-10. This is because adversarial training on CIFAR-10 is a more challenging problem that may have more misclassified examples during training, and MART can better handle those misclassified examples due to its regularization term in (11). Note that the robustness improvement of MART is not caused by the so-called “obfuscated gradients” (Athalye et al., 2018). This can be verified by two phenomenons: (1) strong test attacks (e.g., PGD\(^{20}\)) have higher success rates (lower accuracies) than weak test attacks (e.g., FGSM), and (2) white-box test attacks have higher success rates than back-box test attacks (comparing Table 2 with Table 3). Besides, we conduct an additional check using a gradient-free attack SPSA (Uesato et al., 2018). SPSA attack does not obtain lower accuracy than gradient-based attacks like PGD, which confirms that the robustness of MART trained models are not due to gradient masking.

Black-box Robustness. Black-box test attacks are crafted from the natural test images by attacking a surrogate model with an architecture that is either a copy of the defense model (MNIST) or a more complex ResNet-50 (He et al., 2016) model (CIFAR-10). Both surrogate models are trained separately from the defense models on the original training sets using Standard adversarial training (PGD\(^{10}\) with random start and step size \(\epsilon/4\)). The attacking methods used here are: FGSM, PGD\(^{10}\), PGD\(^{20}\), and CW\(_\infty\). The black-box robustness of all defense models are reported in Table 3. Again, the proposed defense MART achieves higher robustness than other baselines. Compared with the white-box results, all defense methods achieve much better robustness against black-box attacks, even close to the natural accuracy. This suggests that adversarial training is indeed a very practical choice for defense scenarios where the target model can be kept secret from potential attackers. It is also observed that robustness on strong attacks like CW\(_\infty\) is higher than weak attacks like FGSM, which indicates that strong attacks have less transferability than weak attacks (Madry et al., 2018).

4.3 Benchmarking the State-of-the-art Robustness

In this part, we conduct more experiments on a large-capacity network WideResNet (Zagoruyko & Komodakis, 2016) to benchmark the state-of-the-art robustness, and also explore using unlabeled data for further robustness boost.

Performance on WideResNet. We employ WideResNet-34-10 (depth 34 and width 10) to explore the full power of our proposed MART defense method, and also benchmark the state-of-the-art robustness on CIFAR-10. The robustness of all defense models are tested against white-box FGSM,
Table 4: White-box robustness (%) on CIFAR-10 using the WideResNet-34-10.

<table>
<thead>
<tr>
<th>Defense</th>
<th>Natural</th>
<th>FGSM Best</th>
<th>FGSM Last</th>
<th>PGD20 Best</th>
<th>PGD20 Last</th>
<th>PGD100 Best</th>
<th>PGD100 Last</th>
<th>CW∞ Best</th>
<th>CW∞ Last</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>87.30</td>
<td>56.10</td>
<td>56.10</td>
<td>52.68</td>
<td>49.31</td>
<td>51.55</td>
<td>49.03</td>
<td>50.73</td>
<td>48.47</td>
</tr>
<tr>
<td>Dynamic</td>
<td>84.51</td>
<td>63.53</td>
<td>63.53</td>
<td>55.03</td>
<td>51.70</td>
<td>54.12</td>
<td>50.07</td>
<td>51.34</td>
<td>49.27</td>
</tr>
<tr>
<td>TRADES</td>
<td>84.22</td>
<td>64.70</td>
<td>64.70</td>
<td>56.40</td>
<td>53.16</td>
<td>55.68</td>
<td>51.27</td>
<td>51.98</td>
<td>51.12</td>
</tr>
<tr>
<td>MART</td>
<td>84.17</td>
<td><strong>67.51</strong></td>
<td><strong>67.51</strong></td>
<td><strong>58.56</strong></td>
<td><strong>57.39</strong></td>
<td><strong>57.88</strong></td>
<td><strong>55.04</strong></td>
<td><strong>54.58</strong></td>
<td><strong>54.53</strong></td>
</tr>
</tbody>
</table>

Table 5: White-box robustness (%) on WideResNet with additional unlabeled data.

a) WideResNet-34-8 with 100K unlabeled data

<table>
<thead>
<tr>
<th>Defense</th>
<th>Natural</th>
<th>PGD20</th>
</tr>
</thead>
<tbody>
<tr>
<td>UAT++</td>
<td>86.04</td>
<td>59.41</td>
</tr>
<tr>
<td>RST</td>
<td>88.24</td>
<td>59.60</td>
</tr>
<tr>
<td>MART</td>
<td>86.68</td>
<td>61.88</td>
</tr>
</tbody>
</table>

b) WideResNet-28-10 with 500K unlabeled data

<table>
<thead>
<tr>
<th>Defense</th>
<th>Natural</th>
<th>PGD20</th>
</tr>
</thead>
<tbody>
<tr>
<td>UAT++</td>
<td>86.21</td>
<td>62.76</td>
</tr>
<tr>
<td>RST</td>
<td>89.70</td>
<td>63.10</td>
</tr>
<tr>
<td>MART</td>
<td>86.30</td>
<td>65.04</td>
</tr>
</tbody>
</table>

PGD20, PGD100 and CW∞ attacks, under the same settings as Section 4.2. We report the robustness of both the best and the last epoch models obtained during training in Table 4. For each defense method against each attack, the “best” refers to the highest robustness that ever achieved at different checkpoints. Specifically, against FGSM attack, the best model was found at the last epoch (e.g., "best" is also "last") for all defense methods, while against PGD20, PGD100 and CW∞ attacks, the best model was found at the epoch right after the first time learning rate decay (i.e., epoch 76). Our proposed MART outperforms all baseline methods in terms of the robustness of both the best and the last epoch models. Particularly under the most common comparison setting (against PGD20 attacks on CIFAR-10), MART improved ~8% over Standard, and ~4% even over TRADES for the last epoch model. A similar trend of improvement is also observed for the best epoch model results. Considering the worst case accuracies against all attacks, MART still gains ~6% and ~3.5% robustness improvement over Standard and TRADES respectively.

**Boosting with Additional Unlabeled Data.** Here, we evaluate the proposed semi-supervised version of MART and show that it can also benefit from additional unlabeled data and achieves better robustness. Following the exact settings in UAT++ (Uesato et al., 2019) and RST (Carmon et al., 2019), we compare the robustness of MART with them on WideResNet-34-8 and WideResNet-28-10 against PGD20 (FGSM20) respectively (in the same setting as reported in their paper). The dataset is CIFAR-10 with 100K and 500K unlabeled data extracted from the 80 Million Tiny Images dataset (Torralba et al., 2008). As confirmed in Table 5, our proposed defense MART can also benefit from unlabeled data, and further improves the UAT++ and RST defenses, achieving the state-of-the-art robustness. This again verifies the benefit of differentiating misclassified and correctly classified examples for improving robustness, and further demonstrates the superiority of our proposed method.

5 Conclusion and Future Work

In this paper, we investigated the interesting observation that misclassified examples have a recognizable impact on the final robustness of adversarial training, especially for the outer minimization process. Based on this observation, we designed a misclassification aware adversarial risk, which is formulated as adding an misclassification aware regularization to the standard adversarial risk. Following the regularized adversarial risk, we proposed a new defense algorithm, called Misclassification Aware adveRsarial Training (MART), with appropriate surrogate loss functions. Experimental results demonstrated that MART can achieve significantly improved adversarial robustness with respect to the state-of-the-art, and can also achieves better robustness with additional unlabeled data.

In the future, we plan to investigate the effect of differentiation of correctly classified/misclassified training examples in the recently proposed certified/provable robustness framework (Cohen et al., 2019; Salman et al., 2019) and explore the potential improvements brought by the differentiation of training examples.

Explanations about the different results of TRADES (Zhang et al., 2019) are in Appendix B.
REFERENCES


A MART Algorithm

Algorithm 1 Misclassification Aware adversarial Training (MART)

1: **Input:** Training data \( \{x_i, y_i\}_{i=1,...,n} \), outer iteration number \( T_O \), inner iteration number \( T_I \), maximum perturbation \( \epsilon \), step size for inner optimization \( \eta_I \), step size for outer optimization \( \eta_O \)
2: **Initialization:** Standard random initialization of \( h_\theta \)
3: for \( \ell = 1, \ldots, T_O \) do
4: Uniformly sample a minibatch of training data \( B^{(\ell)} \)
5: for \( x_i \in B^{(\ell)} \) do
6: \( x'_i = x_i + \epsilon \cdot \xi \), with \( \xi \sim U(-1, 1) \) # \( U \) is a uniform distribution
7: for \( s = 1, \ldots, T_I \) do
8: \( x'_i \leftarrow \Pi_{B,x_i}(x'_i + \eta_I \cdot \text{sign}(\nabla_{x'_i} \text{CE}(p(x'_i, \theta), y_i))) \) # \( \Pi \) is the projection operator
9: end for
10: \( x'_i \leftarrow x'_i \)
11: end for
12: \( \theta \leftarrow \theta - \eta_O \sum_{x_i \in B^{(\ell)}} \nabla_\theta \mathcal{L}(x_i, y_i, x'_i; \theta) \)
13: end for
14: **Output:** Robust classifier \( h_\theta \)

B Explanations about the results of TRADES

To make a fair comparison with the latest method TRADES (Zhang et al., 2019), our code is built upon the TRADES framework. However, TRADES has replication issues, as pointed out by Carmon et al. (2019) and other researchers at GitHub issue #9 “Robust Accuracy on CIFAR10”. The authors of TRADES confirmed that the robustness reported in their paper is the *best* robustness that ever occurred during training. To avoid questioning of our results, we have reported both the *best* and the *last* results in Table 4 and the *best* results for TRADES match their original paper.