COST-SENSITIVE ROBUSTNESS AGAINST ADVERSARIAL EXAMPLES

Anonymous authors
Paper under double-blind review

ABSTRACT

Several recent works have developed methods for training classifiers that are certifiably robust against norm-bounded adversarial perturbations. However, these methods assume that all the adversarial transformations provide equal value for adversaries, which is seldom the case in real-world applications. We advocate for *cost-sensitive robustness* as the criteria for measuring the classifier's performance for specific tasks. We encode the potential harm of different adversarial transformations in a cost matrix, and propose a general objective function to adapt the robust training method of Wong & Kolter (2018) to optimize for cost-sensitive robustness. Our experiments on simple MNIST and CIFAR10 models and a variety of cost matrices show that the proposed approach can produce models with substantially reduced cost-sensitive robust error, while maintaining classification accuracy.

1 INTRODUCTION

Despite the exceptional performance of deep neural networks (DNNs) on various machine learning tasks such as malware detection (Saxe & Berlin, 2015), face recognition (Parkhi et al., 2015) and autonomous driving (Bojarski et al., 2016), recent studies (Szegedy et al., 2014; Goodfellow et al., 2015) have shown that deep learning models are vulnerable to misclassifying inputs, known as *adversarial examples*, that are crafted with targeted but visually-imperceptible perturbations. While several defense mechanisms have been proposed and empirically demonstrated to be successful against existing particular attacks (Papernot et al., 2016; Goodfellow et al., 2015), new attacks (Carlini & Wagner, 2017; Tramèr et al., 2017; Athalye et al., 2018) are repeatedly found that circumvent such defenses. To end this arm race, recent works (Wong & Kolter, 2018; Raghunathan et al., 2018; Wong et al., 2018; Wang et al., 2018) propose methods to certify examples to be robust against some specific norm-bounded adversarial perturbations for given inputs and to train models to optimize for certifiable robustness.

However, all of the aforementioned methods aim at improving the *overall* robustness of the classifier. This means that the methods to improve robustness are designed to prevent seed examples in any class from being misclassified as any other classes. Achieving such a goal requires producing a perfect classifier, and has, unsurprisingly, remained elusive. Indeed, Mahloujifar et al. (2018) proved that if the metric probability space is concentrated, overall adversarial robustness is unattainable for any classifier with initial constant error. We argue that overall robustness may not be the appropriate criteria for measuring system performance in security-sensitive applications, since only certain kinds of adversarial misclassifications pose meaningful threats or provide value for adversaries. Whereas overall robustness places equal emphasis on every adversarial transformation, from a security perspective, only certain transformations matter. As a simple example, misclassifying a malicious program as benign results in more severe consequences than the reverse.

In this paper, we propose a general method for adapting provable defenses against norm-bounded perturbations to take into account the potential harm of different adversarial class transformations. Inspired by cost-sensitive learning (Domingos, 1999; Elkan, 2001) for non-adversarial contexts, we capture the impact of different adversarial class transformations using a cost matrix $\boldsymbol{C}$, where each entry represents the cost of an adversary being able to take a natural example from the first class and perturb it so as to be misclassified by the model as the second class. Instead of reducing the overall robust error, our goal is to minimize the cost-weighted robust error (which we define for both binary and real-valued costs in $\boldsymbol{C}$). The proposed method incorporates the specified cost matrix into the

training objective function, which encourages stronger robustness guarantees on cost-sensitive class transformations, while maintaining the overall classification accuracy on the original inputs.

Contributions. We introduce the notion of cost-sensitive robustness (Section 3.1) as a criteria to assess the expected performance of a classifier when facing adversarial examples. Specifically, by encoding the consequences of different adversarial transformations into a cost matrix, we propose an objective function for training a cost-sensitive robust classifier (Section 3.2) for any given task. The proposed method is general in that it can incorporate any type of cost matrix, including both binary and real-valued. We demonstrate the effectiveness of the proposed cost-sensitive defense model for a variety of cost scenarios on two benchmark image classification datasets: MNIST (Section 4.1) and CIFAR10 (Section 4.2). Compared with the state-of-the-art overall robust defense model (Wong & Kolter, 2018), our model achieves significant improvements in cost-sensitive robustness for different tasks, while maintaining approximately the same classification accuracy on both datasets.

Notation. We use lower-case boldface letters such as $\boldsymbol{x}$ for vectors and capital boldface letters such as $\boldsymbol{A}$ to represent matrices. Let $[m]$ be the index set $\{1, 2, \ldots, m\}$ and A_{ij} be the (i, j)-th entry of matrix $\boldsymbol{A}$. Denote the i-th natural basis vector, the all-ones vector and the identity matrix by $\boldsymbol{e}_i$, $\mathbf{1}$ and $\boldsymbol{I}$ respectively. For any vector $\boldsymbol{x} \in \mathbb{R}^d$, the ℓ_∞-norm of $\boldsymbol{x}$ is defined as $\|\boldsymbol{x}\|_\infty = \max_{i\in[d]} |x_i|$.

2 Background

In this section, we provide a brief introduction on related topics, including neural network classifiers, adversarial examples, defenses with certified robustness, and cost-sensitive learning.

2.1 Neural Network Classifiers

A K-layer neural network classifier can be represented by a function $f : \mathcal{X} \to \mathcal{Y}$ such that $f(\boldsymbol{x}) = f_{K-1}(f_{K-2}(\cdots(f_1(\boldsymbol{x}))))$, for any $\boldsymbol{x} \in \mathcal{X}$. For $k \in \{1, 2, \ldots, K-2\}$, the mapping function $f_k(\cdot)$ typically consists of two operations: an affine transformation (either matrix multiplication or convolution) and a nonlinear activation. In this paper, we consider rectified linear unit (ReLU) as the activation function. If denote the feature vector of the k-th layer as $\boldsymbol{z}_k$, then $f_k(\cdot)$ is defined as

$$\boldsymbol{z}_{k+1} = f_k(\boldsymbol{z}_k) = \max\{\boldsymbol{W}_k \boldsymbol{z}_k + \boldsymbol{b}_k, \mathbf{0}\}, \quad \forall k \in \{1, 2, \ldots K-2\},$$

where $\boldsymbol{W}_k$ denotes the weight parameter matrix and $\boldsymbol{b}_k$ the bias vector. The output function $f_{K-1}(\cdot)$ maps the feature vector in the last hidden layer to the output space $\mathcal{Y}$ solely through matrix multiplication: $\boldsymbol{z}_K = f_{K-1}(\boldsymbol{z}_{K-1}) = \boldsymbol{W}_{K-1}\boldsymbol{z}_{K-1} + \boldsymbol{b}_{K-1}$, where $\mathbf{z}_K$ can be regarded as the estimated score vector of input $\boldsymbol{x}$ for different possible output classes. In the following discussions, we use f_θ to represent the neural network classifier, where $\theta = \{\boldsymbol{W}_1, \ldots, \boldsymbol{W}_{K-1}, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_{K-1}\}$ denotes the model parameters.

To train the neural network, a loss function $\sum_{i=1}^N \mathcal{L}(f_\theta(\boldsymbol{x}_i), y_i)$ is defined for a set of training examples $\{\boldsymbol{x}_i, y_i\}_{i=1}^N$, where $\boldsymbol{x}_i$ is the i-th input vector and y_i denotes its class label. Cross-entropy loss is typically used for multiclass image classification. With proper initialization, all model parameters are then updated iteratively using backpropagation. For any input example $\widetilde{\boldsymbol{x}}$, the predicted label $\widehat{y}$ is given by the index of the largest predicted score among all classes, $\operatorname{argmax}_j [f_\theta(\widetilde{\boldsymbol{x}})]_j$.

2.2 Adversarial Examples

An adversarial example is an input, generated by some adversary, which is visually indistinguishable from an example from the natural distribution, but is able to mislead the target classifier. Since "visually indistinguishable" depends on human perception, which is hard to define rigorously, we consider the most popular alternative: input examples with perturbations bounded in ℓ_∞-norm (Goodfellow et al., 2015). More formally, the set of adversarial examples with respect to seed example $\{\boldsymbol{x}_0, y_0\}$ and classifier $f_\theta(\cdot)$ is defined as

$$\mathcal{A}_\epsilon(\boldsymbol{x}_0, y_0; \theta) = \big\{\boldsymbol{x} \in \mathcal{X} : \|\boldsymbol{x} - \boldsymbol{x}_0\|_\infty \le \epsilon \text{ and } \operatorname*{argmax}_j [f_\theta(\boldsymbol{x})]_j \neq y_0\big\}, \tag{2.1}$$

where $\epsilon > 0$ denotes the maximum perturbation distance. Although ℓ_p distances are commonly used in adversarial examples research, they are not an adequate measure of perceptual similarity (Sharif

et al., 2018) and other minimal geometric transformations can be used to find adversarial examples (Engstrom et al., 2017; Kanbak et al., 2017; Xiao et al., 2018). Nevertheless, there is considerable interest in improving robustness in this simple domain, and hope that as this research area matures we will find ways to apply results from studying simplified problems to more realistic ones.

2.3 Defenses with Certified Robustness

A line of recent work has proposed defenses that are guaranteed to be robust against norm-bounded adversarial perturbations. Hein & Andriushchenko (2017) proved formal robustness guarantees against ℓ_2-norm bounded perturbations for two-layer neural networks, and provided a training method based on a surrogate robust bound. Raghunathan et al. (2018) developed an approach based on semidefinite relaxation for training certified robust classifiers, but was limited to two-layer fully-connected networks. Our work builds most directly on Wong & Kolter (2018), which can be applied to deep ReLU-based networks and achieves the state-of-the-art certified robustness on MNIST dataset.

Following the definitions in Wong & Kolter (2018), an adversarial polytope $\mathcal{Z}_\epsilon(\boldsymbol{x})$ with respect to a given example $\boldsymbol{x}$ is defined as

$$\mathcal{Z}_\epsilon(\boldsymbol{x}) = \big\{ f_\theta(\boldsymbol{x} + \boldsymbol{\Delta}) : \|\boldsymbol{\Delta}\|_\infty \leq \epsilon \big\}, \tag{2.2}$$

which contains all the possible output vectors for the given classifier f_θ by perturbing $\boldsymbol{x}$ within an ℓ_∞-norm ball with radius ϵ. A seed example, $\{\boldsymbol{x}_0, y_0\}$, is said to be *certified robust* with respect to maximum perturbation distance ϵ, if the corresponding adversarial example set $\mathcal{A}_\epsilon(\boldsymbol{x}_0, y_0; \theta)$ is empty. Equivalently, if we solve, for any output class $y_{\text{targ}} \neq y_0$, the optimization problem,

$$\underset{\boldsymbol{z}_K}{\text{minimize}} \;\; [\boldsymbol{z}_K]_{y_0} - [\boldsymbol{z}_K]_{y_{\text{targ}}}, \quad \text{subject to} \;\; \boldsymbol{z}_K \in \mathcal{Z}_\epsilon(\boldsymbol{x}_0), \tag{2.3}$$

then according to the definition of $\mathcal{A}_\epsilon(\boldsymbol{x}_0, y_0; \theta)$ in (2.1), $\{\boldsymbol{x}_0, y_0\}$ is guaranteed to be robust provided that the optimal objective value of (2.3) is positive for every output class. To train a robust model on a given dataset $\{\boldsymbol{x}_i, y_i\}_{i=1}^N$, the standard robust optimization aims to minimize the sample loss function on the worst-case locations through the following adversarial loss

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{N} \max_{\|\boldsymbol{\Delta}\|_\infty \leq \epsilon} \mathcal{L}\big(f_\theta(\boldsymbol{x}_i + \boldsymbol{\Delta}), y_i\big), \tag{2.4}$$

where $\mathcal{L}(\cdot, \cdot)$ denotes the cross-entropy loss. However, due to the nonconvexity of the neural network classifier $f_\theta(\cdot)$ introduced by the nonlinear ReLU activation, both the adversarial polytope (2.2) and training objective (2.4) are highly nonconvex. In addition, solving optimization problem (2.3) for each pair of input example and output class is computationally intractable.

Instead of solving the optimization problem directly, Wong & Kolter (2018) proposed an alternative training objective function based on convex relaxation, which can be efficiently optimized through a dual network. Specifically, they relaxed $\mathcal{Z}_\epsilon(\boldsymbol{x})$ into a convex outer adversarial polytope $\widetilde{\mathcal{Z}}_\epsilon(\boldsymbol{x})$ by replacing the ReLU inequalities for each neuron $z = \max\{\widehat{z}, 0\}$ with a set of inequalities,

$$z \geq 0, \quad z \geq \widehat{z}, \quad -u\widehat{z} + (u - \ell)z \leq -u\ell, \tag{2.5}$$

where u, ℓ denote the lower and upper bounds on the considered pre-ReLU activation.[1] Based on the relaxed outer bound $\widetilde{\mathcal{Z}}_\epsilon(\boldsymbol{x})$, they propose the following alternative optimization problem,

$$\underset{\boldsymbol{z}_K}{\text{minimize}} \;\; [\boldsymbol{z}_K]_{y_0} - [\boldsymbol{z}_K]_{y_{\text{targ}}}, \quad \text{subject to} \;\; \boldsymbol{z}_K \in \widetilde{\mathcal{Z}}_\epsilon(\boldsymbol{x}_0), \tag{2.6}$$

which is in fact a linear program. Since $\mathcal{Z}_\epsilon(\boldsymbol{x}) \subseteq \widetilde{\mathcal{Z}}_\epsilon(\boldsymbol{x})$ for any $\boldsymbol{x} \in \mathcal{X}$, solving (2.6) for all output classes provides stronger robustness guarantees compared with (2.3), provided all the optimal objective values are positive. In addition, they derived a guaranteed lower bound, denoted by $J_\epsilon\big(\boldsymbol{x}_0, g_\theta(\boldsymbol{e}_{y_0} - \boldsymbol{e}_{y_{\text{targ}}})\big)$, on the optimal objective value of (2.6) using duality theory, where $g_\theta(\cdot)$ is a K-layer feedforward dual network (Theorem 1 in Wong & Kolter (2018)). Finally, according to the properties of cross-entropy loss, they minimize the following objective to train the robust model, which serves as an upper bound of the adversarial loss (2.4):

$$\underset{\theta}{\text{minimize}} \;\; \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\bigg(-J_\epsilon\big(\boldsymbol{x}_i, g_\theta(\boldsymbol{e}_{y_i} \cdot \mathbf{1}^\top - \boldsymbol{I})\big), y_i \bigg), \tag{2.7}$$

[1] The elementwise activation bounds can be computed efficiently using Algorithm 1 in Wong & Kolter (2018).

where $g_\theta(\cdot)$ is regarded as a columnwise function when applied to a matrix. Although the proposed method in Wong & Kolter (2018) achieves certified robustness, its computational complexity is quadratic with the network size in the worst case so it only scales to small networks. Recently, Wong et al. (2018) extended the training procedure to scale to larger networks by using nonlinear random projections. However, if the network size allows for both methods, we observe a small decrease in performance using the training method provided in Wong et al. (2018). Therefore, we only use the approximation techniques for the experiments on CIFAR10 (§4.2), and use the less scalable method for the MNIST experiments (§4.1).

2.4 Cost-Sensitive Learning

Cost-sensitive learning (Domingos, 1999; Elkan, 2001; Liu & Zhou, 2006) was proposed to deal with unequal misclassification costs and class imbalance problem in many real-world applications, such as database marketing, fraud detection and medical diagnosis. The key observation is that cost-blind learning algorithms tend to overwhelm the major class, but the neglected minor class is often our primary interest. For example, in medical diagnosis misclassifying a rare cancerous lesion as benign is extremely costly. Various cost-sensitive learning algorithms (Kukar & Kononenko, 1998; Zadrozny et al., 2003; Zhou & Liu, 2010) has been proposed in literature, but few of them considered adversarial settings. Dalvi et al. (2004) studied the naive Bayes classifier for spam detection in the presence of a cost-sensitive adversary, and developed an adversary-aware classifier based on game theory. Asif et al. (2015) proposed a cost-senstive robust minimax approach that hardens a linear discriminant classifier with robustness in the adversarial context. All of these methods are developed for simple linear classifiers, which cannot be extended to neural network classifiers directly. In addition, the robustness of their proposed classifier is only examined experimentally based on the performance against some specific adversary, so does not provide any notion of certified robustness. Recently, Dreossi et al. (2018) advocated for the idea of using application-level semantics in adversarial analysis, however, they didn't provide a formal method on how to train such classifier. In contrast, our work provides a practical training method that hardens neural network classifiers with certified cost-sensitive robustness against adversarial perturbations.

3 Training Cost-Sensitive Robust Classifier

The approach introduced in Wong & Kolter (2018) penalizes all adversarial class transformations equally, even though the consequences of adversarial examples usually depends on the specific class transformations. Here, we provide a formal definition of cost-sensitive robustness (§3.1) and propose a general method for training cost-sensitive robust models (§3.2).

3.1 Certified Cost-Sensitive Robustness

Our approach uses a cost matrix $\boldsymbol{C}$ that encodes the cost of different adversarial examples. First, we consider the case where there are m classes and $\boldsymbol{C}$ is a $m \times m$ binary matrix with $C_{i,j} \in \{0, 1\}$. The value $C_{jj'}$ indicates whether we care about an adversary transforming a seed input in class j into one recognized by the model as being in class j'. If the adversarial transformation $j \to j'$ matters, $C_{jj'} = 1$, otherwise $C_{jj'} = 0$. Let $\Omega_j = \{j' \in [m] : C_{jj'} \neq 0\}$ be the index set of output classes that induce cost with respect to input class j. For any $j \in [m]$, let $\delta_j = 0$ if Ω_j is an empty set, and $\delta_j = 1$ otherwise. We are only concerned with adversarial transformations from a seed class j to target classes $j' \in \Omega_j$. For any example $\boldsymbol{x}$ in seed class j, $\boldsymbol{x}$ is said to be *certified cost-sensitive robust* if the lower bound $J_\epsilon(\boldsymbol{x}, g_\theta(\boldsymbol{e}_j - \boldsymbol{e}_{j'})) \geq 0$ for all $j' \in \Omega_j$. That is, no adversarial perturbations in an ℓ_∞-norm ball around $\boldsymbol{x}$ with radius ϵ can mislead the classifier to any target class in Ω_j. The *cost-sensitive robust error* on a dataset $\{\boldsymbol{x}_i, y_i\}_{i=1}^N$ is defined as the number of examples that are not guaranteed to be cost-sensitive robust over the total number of valued seed examples:

$$\textit{cost-sensitive robust error} = 1 - \frac{\#\{i \in [N] : J_\epsilon(\boldsymbol{x}_i, g_\theta(\boldsymbol{e}_{y_i} - \boldsymbol{e}_{j'})) \geq 0, \forall j' \in \Omega_{y_i}\}}{\sum_{j|_{\delta_j=1}} N_j},$$

where $\#\mathcal{A}$ represents the cardinality of a set $\mathcal{A}$, and N_j is the total number of examples in class j.

Next, we consider a more general case where $\boldsymbol{C}$ is a $m \times m$ real-valued cost matrix. Each entry of $\boldsymbol{C}$ is a non-negative real number, which represents the cost of the corresponding adversarial

transformation. To take into account the different potential costs among adversarial examples, we propose to measure the cost-sensitive robustness by the average certified cost of adversarial examples. The cost of an adversarial example $\boldsymbol{x}$ in class j is defined as the sum of all $C_{jj'}$ such that $J_\epsilon(\boldsymbol{x}, g_\theta(\boldsymbol{e}_j - \boldsymbol{e}_{j'})) < 0$. Intuitively speaking, an adversarial example will induce more cost if it can be adversarially misclassified as more target classes with high cost. Accordingly, the *robust cost* is defined as the total cost of adversarial examples divided by the total number of valued seed examples:

$$\textit{robust cost} = \frac{\sum_{j|_{\delta_j=1}} \sum_{i|_{y_i=j}} \sum_{j'\in\Omega_j} C_{jj'} \cdot \mathbb{1}\left(J_\epsilon(\boldsymbol{x}_i, g_\theta(\boldsymbol{e}_j - \boldsymbol{e}_{j'})) < 0\right)}{\sum_{j|_{\delta_j=1}} N_j}, \tag{3.1}$$

where $\mathbb{1}(\cdot)$ denotes the indicator function.

3.2 Cost-Sensitive Robust Optimization

Recall that our goal is to develop a classifier with certified cost-sensitive robustness as defined in §3.1, while maintaining overall classification accuracy. According to the guaranteed lower bound, $J_\epsilon\big(\boldsymbol{x}_0, g_\theta(\boldsymbol{e}_{y_0} - \boldsymbol{e}_{y_{\text{targ}}})\big)$ on (2.6), we propose to solve the following robust optimization problem with respect to a neural network classifier f_θ:

$$\begin{aligned}\underset{\theta}{\text{minimize}}\ & \frac{1}{N}\sum_{i\in[N]} \mathcal{L}\big(f_\theta(\boldsymbol{x}_i), y_i\big) \\ & + \alpha \sum_{j\in[m]} \frac{\delta_j}{N_j} \sum_{i|_{y_i=j}} \log\left(1 + \sum_{j'\in\Omega_j} C_{jj'} \cdot \exp\big(-J_\epsilon(\boldsymbol{x}_i, g_\theta(\boldsymbol{e}_j - \boldsymbol{e}_{j'}))\big)\right),\end{aligned} \tag{3.2}$$

where $\alpha \geq 0$ denotes the regularization parameter. The first term in (3.2) denotes the cross-entropy loss for standard classification, whereas the second term accounts for the cost-sensitive robustness. Compared with the overall robustness training objective function (2.7), we include a regularization parameter α to control the trade-off between classification accuracy on original inputs and adversarial robustness. To provide cost-sensitivity, the loss function selectively penalizes the adversarial examples based on their cost. For binary cost matrixes, the regularization term penalizes every cost-sensitive adversarial example equally, but has no impact for instances where $C_{jj'} = 0$. For the real-valued costs, a larger value of $C_{jj'}$ increases the weight of the corresponding adversarial transformation in the training objective. Optimization problem (3.2) can be solved efficiently using gradient-based algorithms, such as stochastic gradient descent and ADAM (Kinga & Adam, 2015).

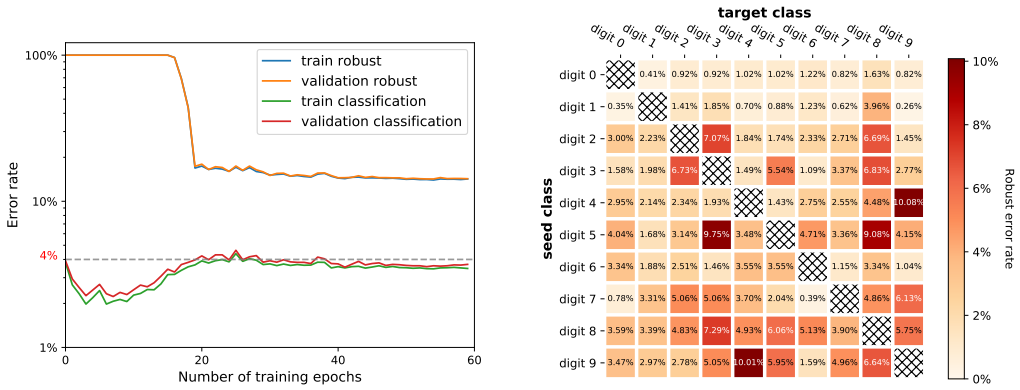
4 Experiments

We evaluate the performance of the proposed cost-sensitive robustness training on models for two benchmark image classification datasets: MNIST (LeCun et al., 2010) and CIFAR10 (Krizhevsky & Hinton, 2009). We compare our results for various cost scenarios with overall robustness training (§2.3) as a baseline. For both datasets, the family of adversarial attacks is specified as all the adversarial perturbations that are bounded in an ℓ_∞-norm ball.

4.1 MNIST

We use the same convolutional neural network architecture (LeCun et al., 1998) for MNIST as Wong & Kolter (2018), which includes two convolutional layers, with 16 and 32 filters respectively, and a two fully-connected layers, consisting of 100 and 10 hidden units respectively. ReLU activations are applied to each layer expect the last one. For both our cost-sensitive robust model and the overall robust model, we randomly split the 60,000 training samples into five folds of equal size, train the classifier over 60 epochs on four of them using the Adam optimizer (Kinga & Adam, 2015) with batch size 50 and learning rate 0.001, and treat the remaining one as validation dataset for model selection. In addition, we use the ϵ-scheduling and learning rate decay techniques, where we increase ϵ from 0.05 to the desired value linearly over the first 20 epochs and decay the learning rate by 0.5 every 10 epochs for the remaining epochs.

Baseline: Overall Robustness. Figure 1(a) illustrates the learning curves of both classification error and overall robust error during training based on robust loss (2.7) with maximum perturbation



(a) learning curves (b) heatmap of robust test error

Figure 1: Preliminary results on MNIST using overall robust classifier: (a) learning curves of the classification error and overall robust error over the 60 training epochs; (b) heatmap of the robust test error for pairwise class transformations based on the best trained classifier.

distance $\epsilon = 0.2$. The model with classification error less than 4% and minimum overall robust error on the validation dataset is selected over the 60 training epochs. The best classifier reaches 3.39% classification error and 13.80% overall robust error on the 10,000 MNIST testing samples. We report the robust test error for every adversarial transformation in Figure 1(b) (for the model without any robustness training all of the values are 100%). The (i, j)-th entry is a bound on the robustness of that seed-target transformation—the fraction of testing examples in class i that cannot be certified robust against transformation into class j for any ϵ norm-bounded attack. As shown in Figure 1(b), the attack vulnerability differs considerably among class pairs. For instance, only 0.26% of seeds in class 1 cannot be certified robust for target class 9 compare to 10% of seeds from class 9 into class 4.

Binary Cost Matrix. Next, we evaluate the effectiveness of cost-sensitive robustness training in producing models that are more robust for the important adversarial transformations. We consider four types of tasks defined by different binary cost matrices that capture different sets of adversarial transformations: *single pair*: particular seed class s to particular target class t; *single seed*: particular seed class s to any target class; *single target*: any seed class to particular target class t; and *multiple*: multiple seed and target classes. For each setting, the cost matrix is defined as $\boldsymbol{C}_{i,j} = 1$ if (i, j) is selected; $\boldsymbol{C}_{i,j} = 0$ otherwise. In general, we expect that the sparser the cost matrix, the more opportunity there is for cost-sensitive robustness to improve on overall robustness.

For the single pair task, we selected three representative adversarial goals based on the original robustness in Figure 1(b): a low vulnerability pair (0,2), medium vulnerability pair (6,5) and high vulnerability pair (4,9). Similarly, for the single seed and single target tasks we select three examples representing low, medium, and high vulnerability (see Appendix B.1 for results for all classes). For the multiple task, we consider four variations: (i) the ten most vulnerable seed-target transformations; (ii) ten randomly-selected seed-target transformations; (iii) all the class transformations from odd digit seed to any other class; (iv) all the class transformations from even digit seed to any other class.

Table 1 summarizes the results, comparing the cost-sensitive robust error between the baseline model trained for overall robustness and a model trained using our cost-sensitive robust optimization. The proposed cost-sensitive robust defense model is trained with $\epsilon = 0.2$ based on loss function (3.2) and the corresponding cost matrix $\boldsymbol{C}$ (see Appendix B.2 for comparison results with different ϵ). The regularization parameter α is tuned via cross validation (see Appendix A in the supplementary materials for details). We report the selected best α, classification error and cost-sensitive robust error on the testing dataset.

Our model achieves a substantial improvement on the cost-sensitive robustness compared with the baseline model on all of the considered tasks. The decrease on cost-sensitive robust error varies from 30% to 90%, and is generally higher for lower sparsity cost matrices. In particular, our classifier

Table 1: Comparisons between different robust defense models on MNIST dataset against ℓ_∞ norm-bounded adversarial perturbations with $\epsilon = 0.2$. The sparsity gives the number of non-zero entries in the cost matrix over the total number of possible adversarial transformations. The candidates column is the number of potential seed examples for each task.

Task Description		**Sparsity**	**Candidates**	**Best α**	**Classification Error**		**Robust Error**	
					baseline	ours	baseline	ours
single pair	(0,2)	1/90	980	10.0	3.39%	2.68%	0.92%	0.31%
	(6,5)	1/90	958	5.0	3.39%	2.49%	3.55%	0.42%
	(4,9)	1/90	982	4.0	3.39%	3.00%	10.08%	1.02%
single seed	digit 0	9/90	980	10.0	3.39%	3.48%	3.67%	0.92%
	digit 2	9/90	1032	1.0	3.39%	2.91%	14.34%	3.68%
	digit 8	9/90	974	0.4	3.39%	3.37%	22.28%	5.75%
single target	digit 1	9/90	8865	4.0	3.39%	3.29%	2.23%	0.14%
	digit 5	9/90	9108	2.0	3.39%	3.24%	3.10%	0.29%
	digit 8	9/90	9026	1.0	3.39%	3.52%	5.24%	0.54%
multiple	top 10	10/90	6024	0.4	3.39%	3.34%	11.14%	7.02%
	random 10	10/90	7028	0.4	3.39%	3.18%	5.01%	2.18%
	odd digit	45/90	5074	0.2	3.39%	3.30%	14.45%	9.97%
	even digit	45/90	4926	0.1	3.39%	2.82%	13.13%	9.44%

Table 2: Comparison results of different robust defense models for tasks with real-valued cost matrix.

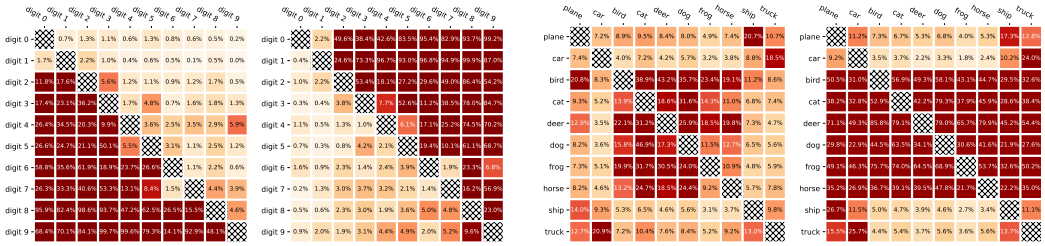
Dataset	**Task**	**Sparsity**	**Candidates**	**Best α**	**Classification Error**		**Robust Cost**	
					baseline	ours	baseline	ours
MNIST	small-large	45/90	10000	0.04	3.39%	3.47%	2.245	0.947
MNIST	large-small	45/90	10000	0.04	3.39%	3.13%	3.344	1.549
CIFAR	vehicle	40/90	4000	0.1	31.80%	26.19%	4.183	3.095

reduces the number of cost-sensitive adversarial examples from 198 to 12 on the single target task with digit 1 as the concerned class.

Real-valued Cost Matrices. Loosely motivated by an forging adversary who obtains value by changing the semantic interpretation of a number, we consider two real-valued cost matrices: *small-large*, where only adversarial transformations from a smaller digit class to a larger one are valued, and the cost of valued-transformation is quadratic with the absolute difference between the seed and target class digits: $C_{ij} = 0$ if $i \geq j$; $C_{ij} = (i-j)^2$ otherwise; *large-small*: only adversarial transformations from a larger digit class to a smaller one are valued: $C_{ij} = 0$ if $i \leq j$; $C_{ij} = (i-j)^2$ otherwise. We tune α for the cost-sensitive robust model on the training MNIST dataset via cross validation, and set all the other parameters the same as in the binary case. The certified robust error for every adversarial transformation on MNIST testing dataset are shown in Figures 2(a) and 2(b), and the classification error and robust cost are given in Table 2. Compared with the heatmap of robust test error based on the overall robust model in Figure 1(b), our trained classifier achieves stronger robustness guarantees on the adversarial transformations that induce costs, especially on those with larger costs.

4.2 CIFAR10

We use the same neural network architecture for the CIFAR10 dataset as Wong et al. (2018), with four convolutional layers and two fully-connected layers. For memory and computational efficiency, we incorporate the approximation technique based on nonlinear random projection during the training phase (Wong et al. (2018), §3.2). We train both the baseline model and our model using random projection of 50 dimensions, and optimize the training objective using SGD. Other parameters such as learning rate and batch size are set as same as those in Wong et al. (2018). Given a specific task,



(a) MNIST (b) MNIST (c) CIFAR10 (d) CIFAR10

Figure 2: Heatmaps of robust test error for pairwise adversarial transformations on: (a) MNIST using our model for *small-large* real-value cost task; (b) MNIST using our model for *large-small* task; (c) CIFAR using the baseline model; (d) CIFAR using our model for real-valued task.

Table 3: Cost-sensitive robust models for CIFAR10 dataset against adversarial attack with $\epsilon = 2/255$.

Task Description		Sparsity	Candidates	Best α	Classification Error		Robust Error	
					baseline	ours	baseline	ours
single pair	(frog, bird)	1/90	1000	10.0	31.80%	27.88%	19.90%	1.20%
	(cat, plane)	1/90	1000	10.0	31.80%	28.63%	9.30%	2.60%
single seed	dog	9/90	1000	0.2	31.80%	30.69%	57.20%	28.90%
	truck	9/90	1000	0.8	31.80%	31.55%	35.60%	15.40%
single target	deer	9/90	9000	0.1	31.80%	26.69%	16.99%	3.77%
	ship	9/90	9000	0.1	31.80%	24.80%	9.42%	3.06%
multiple	A-V	24/90	6000	0.1	31.80%	26.65%	16.67%	7.42%
	V-A	24/90	4000	0.2	31.80%	27.60%	12.07%	8.00%

we train the proposed cost-sensitive robust classifier on 80% randomly-selected training examples, and tune the regularization parameter α according to the performance on the remaining examples as validation dataset. The tasks are similar to those for MNIST (§4.1), except for the multiple task we cluster the ten CIFAR10 classes into two large groups: animals and vehicles, and consider the cases where only transformations between an animal class and a vehicle class are sensitive, and the converse. Table 3 shows the comparison results on the testing data based on different robust defense models with $\epsilon = 2/255$ (see Appendix B.2 for comparison results with larger ϵ). For all of the aforementioned tasks, our model substantially reduces the cost-sensitive robust error while keeping a lower classification error than the baseline.

For the real-valued task, we are concerned with adversarial transformations from seed examples in vehicle classes to other target classes. In addition, more cost is placed on transformations from vehicle to animal, which is 10 times larger compared with that from vehicle to vehicle. Figures 2(c) and 2(d) illustrate the pairwise robust test error using overall robust model and the proposed classifier for the aforementioned real-valued task on CIFAR10.

5 Conclusion

By focusing on overall robustness, previous robustness training methods expend a large fraction of the capacity of the network on unimportant transformations. We argue that for most scenarios, adversarial transformations have different costs depending on the seed and target class, so robust training methods should be designed to account for these differences. By incorporating a cost matrix into the training objective, we proposed a general method for producing a cost-sensitive robust classifier, and demonstrate the effectiveness of our method for a variety of cost scenarios on two simple datasets. There remains a large gap between the small models and limited attacker capabilities for which we can achieve certifiable robustness, and the complex models and unconstrained attacks that may be important in practice, but we hope that considering cost-sensitive robustness will be a step towards achieving more realistic robustness goals for important problems.

REFERENCES

Kaiser Asif, Wei Xing, Sima Behpour, and Brian D Ziebart. Adversarial cost-sensitive classification. In *Thirty-First Conference on Uncertainty in Artificial Intelligence*, pp. 92–101, 2015.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, 2018.

Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

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

Nilesh Dalvi, Pedro Domingos, Sumit Sanghai, Deepak Verma, et al. Adversarial classification. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 99–108. ACM, 2004.

Pedro Domingos. Metacost: A general method for making classifiers cost-sensitive. In *Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 155–164. ACM, 1999.

Tommaso Dreossi, Somesh Jha, and Sanjit A Seshia. Semantic adversarial deep learning. *arXiv preprint arXiv:1804.07045*, 2018.

Charles Elkan. The foundations of cost-sensitive learning. In *International Joint Conference on Artificial Intelligence*, volume 17, pp. 973–978. Lawrence Erlbaum Associates Ltd, 2001.

Logan Engstrom, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling CNNs with simple transformations. *arXiv preprint arXiv:1712.02779*, 2017.

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

Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information Processing Systems*, pp. 2266–2276, 2017.

Can Kanbak, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Geometric robustness of deep networks: analysis and improvement. *arXiv preprint arXiv:1711.09115*, 2017.

Salman H Khan, Munawar Hayat, Mohammed Bennamoun, Ferdous A Sohel, and Roberto Togneri. Cost-sensitive learning of deep feature representations from imbalanced data. *IEEE transactions on neural networks and learning systems*, 29(8):3573–3587, 2018.

D Kinga and J Ba Adam. A method for stochastic optimization. In *International Conference on Learning Representations*, volume 5, 2015.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Matjaž Kukar and Igor Kononenko. Cost-sensitive learning with neural networks. In *13th European Conference on Artificial Intelligence*, pp. 445–449, 1998.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2, 2010.

Xu-Ying Liu and Zhi-Hua Zhou. The influence of class imbalance on cost-sensitive learning: An empirical study. In *Sixth International Conference on Data Mining*, pp. 970–974. IEEE, 2006.

Saeed Mahloujifar, Dimitrios I Diochnos, and Mohammad Mahmoody. The curse of concentration in robust learning: Evasion and poisoning attacks from concentration of measure. *arXiv preprint arXiv:1809.03063*, 2018.

Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE Symposium on Security and Privacy*. IEEE, 2016.

Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Deep face recognition. In *British Machine Vision Conference*, volume 1, pp. 6, 2015.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

Joshua Saxe and Konstantin Berlin. Deep neural network based malware detection using two dimensional binary program features. In *10th International Conference on Malicious and Unwanted Software*, pp. 11–20. IEEE, 2015.

Mahmood Sharif, Lujo Bauer, and Michael K Reiter. On the suitability of l_p-norms for creating and preventing adversarial examples. In *CVPR Workshop on Bright and Dark Sides of Computer Vision: Challenges and Opportunities for Privacy and Security*, 2018.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Formal security analysis of neural networks using symbolic intervals. In *USENIX Security Symposium*, 2018.

Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pp. 5283–5292, 2018.

Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J Zico Kolter. Scaling provable adversarial defenses. *arXiv preprint arXiv:1805.12514*, 2018.

Chaowei Xiao, Jun-Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song. Spatially transformed adversarial examples. *arXiv preprint arXiv:1801.02612*, 2018.

Bianca Zadrozny, John Langford, and Naoki Abe. Cost-sensitive learning by cost-proportionate example weighting. In *Third IEEE International Conference on Data Mining*, pp. 435–442. IEEE, 2003.

Zhi-Hua Zhou and Xu-Ying Liu. On multi-class cost-sensitive learning. *Computational Intelligence*, 26(3):232–257, 2010.

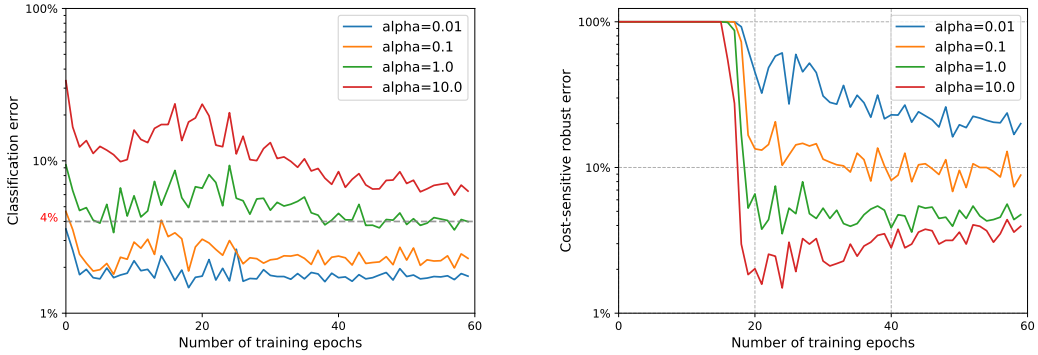
Cost-Sensitive Robustness against Adversarial Examples
Supplemental Materials

A Parameter Tuning

For experiments on the MNIST dataset, we first perform a coarse tuning on regularization parameter α with searching grid $\{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$, and select the most appropriate one, denoted by α_{coarse}, with overall classification error less than 4% and the lowest cost-sensitive robust error on validation dataset. Then, we further finely tune α from the range $\{2^{-3}, 2^{-2}, 2^{-1}, 2^{0}, 2^{1}, 2^{2}, 2^{3}\} \cdot \alpha_{coarse}$, and choose the best robust model according to the same criteria.

Figures 3(a) and 3(b) show the learning curves for task B with digit 9 as the selected seed class based on the proposed cost-sensitive robust model with varying α (we show digit 9 because it is one of the most vulnerable seed classes). The results suggest that as the value of α increases, the corresponding classifier will have a lower cost-sensitive robust error but a higher classification error, which is what we expect from the design of (3.2).

We observe similar trends for the learning curves for the other tasks, so do not present them here. For the CIFAR10 experiments, a similar tuning strategy is implemented. The only difference is that we use 35% as the threshold of overall classification error for selecting the best α.



(a) classification error (b) cost-sensitive robust error

Figure 3: Learning curves for single seed task with digit 9 as the selected seed class on MNIST using the proposed model with varying α: (a) learning curves of classification error; (b) learning curves of cost-sensitive robust error. The maximum perturbation distance is specified as $\epsilon = 0.2$.

B Other Experiments

Here, we provide results from some additional experiments that generally confirm the results included in the paper.

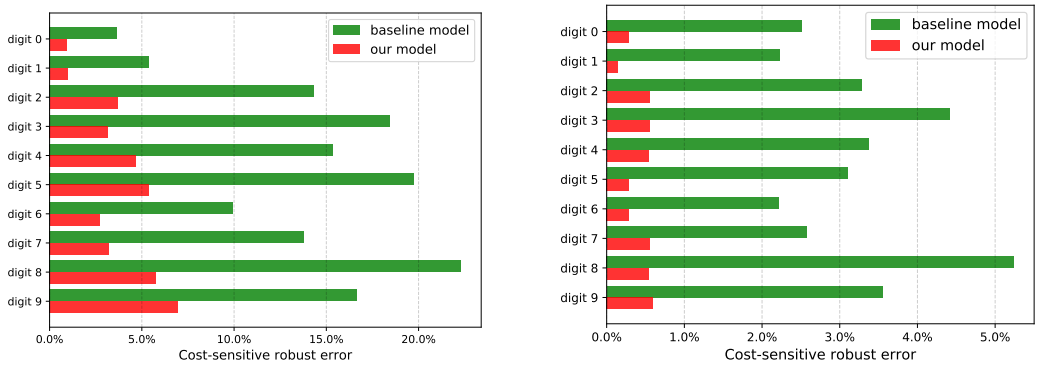
B.1 Single Seed and Target Tasks

In addition to the tasks shown in Table 1, we evaluated all the single-seed and single target tasks for MNIST.

Figures 4(a) and 4(b) demonstrate the comparison results on cost-sensitive robustness between our model and baseline model for each single seed and single target class cost scenario. Our model achieves significant improvements for all of the considered binary tasks. We remark that the overall classification errors of both models are similar in all of the cases, thus we do not present them.

B.2 Varying Adversary Strength

We also investigate the performance of our model against different levels of adversarial strength by varying the value of ϵ that defines the ℓ_∞ ball available to the adversary.

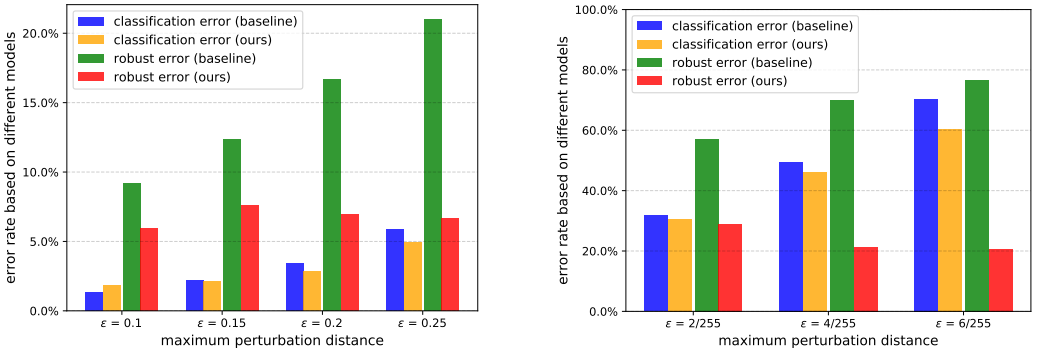


(a) single seed class (b) single target class

Figure 4: Cost-sensitive robust error using the proposed model and baseline model on MNIST for different binary tasks: (a) treat each digit as the seed class of concern respectively; (b) treat each digit as the target class of concern respectively.

Figures 5(a) and 5(b) show the overall classification and cost-sensitive robust error of our best trained model, compared with the baseline model, on MNIST single seed task with digit 9 and CIFAR single seed task with dog as the seed class of concern, respectively. In particular, we vary the maximum ℓ_∞ perturbation distance ϵ from $\{0.1, 0.15, 0.2, 0.25\}$ for MNIST and vary ϵ from $\{2/255, 4/255, 6/255\}$ for CIFAR10.

Under all the considered attack models, the proposed classifier achieves better cost-sensitive adversarial robustness than the baseline, while maintaining similar classification accuracy on original data points. As the adversarial strength increases, the improvement for cost-sensitive robustness over overall robustness becomes more significant.



(a) MNIST (b) CIFAR10

Figure 5: Results for different adversary strengths, ϵ, for different settings: (a) MNIST single seed task with digit 9 as the chosen class; (b) CIFAR10 single seed task with dog as the chosen class.

B.3 Comparison with Standard Cost-Sensitive Classifier

As discussed in Section 2.4, prior work on cost-sensitive learning mainly focuses on the non-adversarial setting. In this section, we investigate the robustness of the cross-entropy based cost-sensitive classifier proposed in Khan et al. (2018), and compare the performance of their classifier with our proposed cost-sensitive robust classifier. Given a set of training examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ and cost matrix C with each entry representing the cost of the corresponding misclassification, the evaluation metric for cost-sensitive learning is defined as the average cost of misclassifications, or

more concretely

$$\textit{misclassfication cost} = \frac{1}{N} \sum_{i \in [N]} C_{y_i \widehat{y}_i}, \quad \text{where } \widehat{y}_i = \operatorname*{argmax}_{j \in [m]} [f_\theta(\mathbf{x}_i)]_j,$$

where m is the total number of class labels and $f_\theta(\cdot)$ denotes the neural network classifier as introduced in Section 2.1. In addition, the cross-entropy based cost-sensitive training objective takes the following form:

$$\underset{\theta}{\text{minimize}} \ \frac{1}{N} \sum_{j \in [m]} \sum_{i|_{y_i = j}} \log \left(1 + \sum_{j' \neq j} C_{jj'} \cdot \exp \left([f_\theta(\mathbf{x}_i)]_{j'} - [f_\theta(\mathbf{x}_i)]_j \right) \right). \tag{B.1}$$

To provide a fair comparison, we assume the cost matrix used for (B.1) coincides with the cost matrix used for cost-sensitive robust training (3.2) in our experiment, whereas they are unlikely to be the same for real security applications[2]. In particular, we consider the small-large real-valued task for MNIST, where the cost matrix C is designed as $C_{ij} = 0.1$, if $i > j$; $C_{ij} = 0$, if $i = j$; $C_{ij} = (i-j)^2$, otherwise. Table B.3 demonstrates the comparison results of different classifiers in such setting: baseline standard deep learning classifier, standard cost-sensitive classifier (Khan et al., 2018) trained using (B.1), classifier trained for overall robustness (Wong & Kolter, 2018) and our proposed classifier trained for cost-sensitive robustness. Compared with baseline, the standard cost-sensitive classifier indeed reduces the misclassification cost, but does not provide any improvement on the robust cost, as defined in (3.1) for real-valued cost matrix. In sharp contrast, our robust training method significantly improves the cost-sensitive robustness.

Table 4: Comparison results of different trained classifiers for small-large real-valued task on MNIST with maximum perturbation distance $\epsilon = 0.2$.

Classifier	**Classification Error**	**Misclassification Cost**	**Robust Cost**
Baseline	0.91%	0.054	85.197
Cost-Sensitive Standard	2.57%	0.016	85.197
Overall Robustness	3.49%	0.252	1.982
Cost-Sensitive Robustness	3.38%	0.060	0.915

[2]For instance, misclassifying a benign program as malicious may still induce some cost in the non-adversarial setting, whereas the adversary may only benefit from transforming a malicious program into a benign one.