Cooperation or Competition: Avoiding Player Domination for Multi-Target Robustness via Adaptive Budgets

Yimu Wang
University of Waterloo
Waterloo, Canada
yimu.wang@uwaterloo.ca

Dinghuai Zhang
Mila, University of Montreal
Montreal, Canada
dinghuai.zhang@mila.quebec

Yihan Wu
University of Pittsburgh
Pittsburgh, United States
yiw154@pitt.edu

Heng Huang
University of Pittsburgh
Pittsburgh, United States
henghuanghh@gmail.com

Hongyang Zhang *
University of Waterloo
Waterloo, Canada
hongyang.zhang@uwaterloo.ca

Abstract

Despite incredible advances, deep learning has been shown to be susceptible to adversarial attacks. Numerous approaches have been proposed to train robust networks both empirically and certifiably. However, most of them defend against only a single type of attack, while recent work takes steps forward in defending against multiple attacks. In this paper, to understand multi-target robustness, we view this problem as a bargaining game in which different players (adversaries) negotiate to reach an agreement on a joint direction of parameter updating. We identify a phenomenon named player domination *in the bargaining game, namely that the existing max-based approaches, such as MAX and MSD, do not converge. Based on our theoretical analysis, we design a novel framework that adjusts the budgets of different adversaries to avoid any player dominance. Experiments on standard benchmarks show that employing the proposed framework to the existing approaches significantly advances multi-target robustness.*

1. Introduction

Machine learning (ML) models [15, 47, 48] have been shown to be susceptible to adversarial examples [39], where human-imperceptible perturbations added to a clean example might arbitrarily change the output of machine learning models. Adversarial examples are generated by maximizing the loss within a small perturbation region around a clean example, *e.g.*, ℓ_∞, ℓ_1 and ℓ_2 balls. On the other hand, numerous heuristic defenses have been proposed to be robust against adversarial examples, *e.g.*, distillation [31], logit-pairing [19] and adversarial training [25].

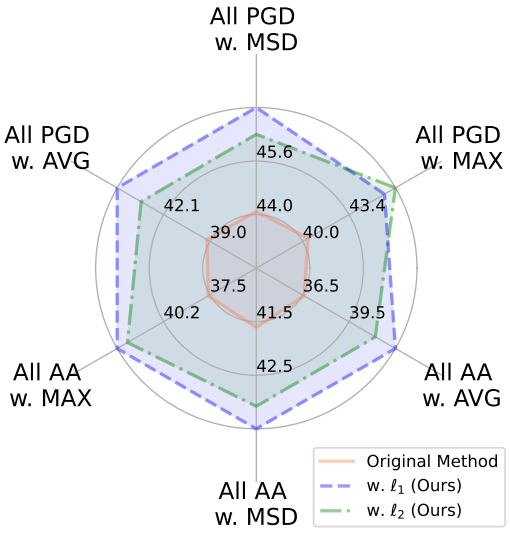


Figure 1. Robust accuracy against PGD attacks and AutoAttack ("AA" in this figure) on CIFAR-10. "All" means that the model successfully defends against the ℓ_1, ℓ_2, and ℓ_∞ (PGD or AutoAttack) attacks simultaneously. Compared with the previously best-known methods, our proposed framework achieves improved performance. "w. ℓ_1" and "w. ℓ_2" refer to the model training with our proposed **AdaptiveBudget** algorithm with ℓ_1 or ℓ_2 norms, respectively.

However, most of the existing defenses are only robust against one type of attacks [11, 25, 33, 49], while they fail to defend against other adversaries. For example, existing work [18, 26] showed that robustness in the ℓ_p threat model does not necessarily generalize to other ℓ_q threat models when $p \neq q$. However, for the sake of the safety of ML systems, it has been argued that one should target robustness against multiple adversaries simultaneously [7].

*Corresponding author.

Recently, various methods [26,35,41] have been proposed to address this problem. Multi-target adversarial training, which targets defending against multiple adversarial perturbations, has attracted significant attention: a variational autoencoder-based model [35] learns a classifier robust to multiple perturbations; after that, MAX and AVG strategies, which aggregate different adversaries for adversarial training against multiple threat models, have been shown to enjoy improved performance [41]. To further advance the robustness against multiple adversaries, MSD [26] is proposed and outperformed MAX and AVG by taking the worst case over all steepest descent directions. These methods follow a general scheme similar to the (single-target) adversarial training. They first sample adversarial examples by different adversaries and then update the model with the aggregation of the gradients from these adversarial examples.

This general scheme for multi-target adversarial training can be seen as an implementation of a cooperative bargaining game [40]. In this game, different parties have to decide how to maximize the surplus they jointly get. In the multi-target adversarial training, we view each party as an adversary, and they negotiate to reach an agreed gradient direction that maximizes the overall robustness.

Inspired by the bargaining game modelling for multi-target adversarial training, we first analyze the convergence property of existing methods, *i.e.*, MAX [41], MSD [26], and AVG [41], and identify a phenomenon namely player domination. Specifically, it refers to the case where one player dominates the bargaining game at any time t, and the gradient at any time t is the same as this player's gradient. Furthermore, we notice that under the SVM and linear model setups, player domination always occurs when using MAX and MSD, which leads to non-convergence. Based on such theoretical results, we propose a novel mechanism that adaptively adjusts the budgets of adversaries to avoid the player domination. We show that with our proposed mechanism, the overall robust accuracy of MAX, AVG and MSD improves on three representative datasets. We also illustrate the performance improvement on CIFAR-10 in Figure 1.

In this paper, we present the first theoretical analysis of the convergence of multi-target robustness on three algorithms under two models. Building on our theoretical results, we introduce a new method called **AdaptiveBudget**, designed to prevent the player domination phenomenon that can cause MSD and MAX to fail to converge. Our extensive experimental results demonstrate the superiority of our approach over previous methods.

2. Related work

Adversarial Training. Goodfellow *et al.* [14] show that even a small perturbation in the direction of the gradient can fool deep learning models for image classification tasks. This is later extended to a multi-step attack [22] called the Basic Iterative Method, now typically referred to as the PGD attack, which significantly improves the success rate of creating adversarial examples. Since then, various variations of the PGD attack [4, 8, 24] have been proposed to overcome heuristic defenses and create stronger adversaries. To defend against these attacks, numerous defense methods [19, 25, 30, 31, 37, 44, 50–56] have been developed. Among these methods, the most successful defense method is adversarial training [25], which formulates the defense problem as a minimax optimization problem and has become one of the few adversarial defenses that is still robust against stronger attacks [1, 5, 27]. As a result, empirical robustness [13, 28, 29, 46, 57] has been significantly advanced over the past few decades.

Multi-target Adversarial Training. Robustness against multiple types of attacks simultaneously is closely related to our work. Schott et al. [35] use multiple variational autoencoders to construct an architecture called "analysis by synthesis" for the MNIST dataset. Their experimental results show that even for MNIST, it is difficult to train a model that is robust to three different adversaries. Following that, Tramer and Boneh [41] investigate the theoretical and empirical trade-offs of adversarial robustness when defending against aggregations of multiple adversaries. Their results show that a model that is robust to the ℓ_∞ adversary might not be able to defend against other attacks, such as ℓ_1 and ℓ_2 attacks, on MNIST. To alleviate this problem, they design an augmentation-based method to achieve ℓ_2 robustness. Later, Croce and Hein [7] propose a provable adversarial defense against all ℓ_p norms for $p \geq 1$ using regularization methods. From a greedy search perspective, Maini et al. [26] suggest that taking the worst-case over all steepest descent directions helps achieve better performance than MAX and AVG empirically. Recently, while not studied as a defense method, Kang et al. [18] investigate the transferability of adversarial robustness between models trained against different perturbation models.

3. Preliminaries

3.1. Problem formulation

The goal of multi-target adversarial training is to learn a function $f_{\mathbf{w}} : \mathcal{X} \to \{-1, +1\}$ that is robust to adversarial examples generated by multiple adversaries[1], where $f_{\mathbf{w}}$ is parameterized by $\mathbf{w}$. The multi-target robust loss of $f_{\mathbf{w}}$ is defined as $\mathbb{E}_{(\mathbf{x},y)}[\max_{\boldsymbol{\delta}\in\mathcal{B}} \ell(f_{\mathbf{w}}(\mathbf{x}+\boldsymbol{\delta}), y)]$, where $\mathcal{B} = \mathcal{B}_1(\epsilon_1) \bigcup \mathcal{B}_2(\epsilon_2) \bigcup \mathcal{B}_\infty(\epsilon_\infty)$, $\mathcal{B}_p(\epsilon) = \{\boldsymbol{\delta} : \|\boldsymbol{\delta}\|_p \leq \epsilon\}$, and $\boldsymbol{\delta}$ is the perturbation. In deep learning scenarios, adversarial training (AT) [25] is frequently used to train a robust classifier. Previous multi-target adversarial training work, *e.g.*, MSD [26], MAX [41], and AVG [41], employ the following

[1] In our paper, we analyze the case where three adversaries are involved, *i.e.*, ℓ_1, ℓ_2 and ℓ_∞.

Algorithm 1 MAX, AVG and MSD algorithms

MAX(input data $\mathbf{x}$, steps k, stepsize η, perturbation budgets $(\epsilon_\infty, \epsilon_1, \epsilon_2)$, loss function ℓ, model $f_{\mathbf{w}}$):
  $\boldsymbol{\delta}_p \leftarrow \mathrm{PGD}(\mathbf{x}, k, \eta, \epsilon_p, \ell, f_{\mathbf{w}}), \forall p \in \{1, 2, \infty\}$;
  Return $\mathrm{argmax}_{\boldsymbol{\delta} \in \{\boldsymbol{\delta}_1, \boldsymbol{\delta}_2, \boldsymbol{\delta}_\infty\}} \ell(f_{\mathbf{w}}(\mathbf{x} + \boldsymbol{\delta}_p), y)$.

AVG(input data $\mathbf{x}$, steps k, stepsize η, perturbation budgets $(\epsilon_\infty, \epsilon_1, \epsilon_2)$, loss function ℓ, model $f_{\mathbf{w}}$):
  Return $\{\mathrm{PGD}(\mathbf{x}, k, \eta, \epsilon_p, \ell, f_{\mathbf{w}})\}_{p \in \{1,2,\infty\}}$.

MSD(input data $\mathbf{x}$, steps k, stepsize η, perturbation budgets $(\epsilon_\infty, \epsilon_1, \epsilon_2)$, loss function ℓ, model $f_{\mathbf{w}}$):
  $\boldsymbol{\delta}^0 = \mathbf{0}$;
  for $i \in [k]$ **do**
    $\boldsymbol{\delta}_p^i \leftarrow \mathrm{PGD}_{\mathrm{Step}}(\mathbf{x}, \boldsymbol{\delta}^i, \eta, \epsilon_p, \ell, f_{\mathbf{w}}), \forall p \in \{1, 2, \infty\}$;
    $\boldsymbol{\delta}^{i+1} \leftarrow \mathrm{argmax}_{\boldsymbol{\delta}_p \in \{\boldsymbol{\delta}_1^i, \boldsymbol{\delta}_2^i, \boldsymbol{\delta}_\infty^i\}} \ell(f_{\mathbf{w}}(\mathbf{x} + \boldsymbol{\delta}_p^i), y)$;
  end for
  Return $\boldsymbol{\delta}^k$.

minimax objective to update the model

$$\min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)} \max_{\boldsymbol{\delta} \in \mathcal{B}} \ell(f_{\mathbf{w}}(\mathbf{x} + \boldsymbol{\delta}), y)\,. \tag{1}$$

This minimax problem is usually decomposed into a two-stage problem with a maximization problem of finding the optimal $\boldsymbol{\delta}$ and a minimization problem of finding the optimal $\mathbf{w}$ given optimal $\boldsymbol{\delta}$, and then iteratively optimizing $\boldsymbol{\delta}$ and $\mathbf{w}$ for several rounds. Under the non-convex scenario, to find the approximate optimal perturbation $\boldsymbol{\delta}$ and the approximate optimal parameter $\mathbf{w}$, gradient descent algorithm [10, 20] and projected gradient descent (PGD) attack are used. Specifically, PGD runs several predefined steps as $\mathrm{PGD}_{\mathrm{step}}(\mathbf{x}, \boldsymbol{\delta}^i, \eta, \ell, \epsilon_p, f_{\mathbf{w}}) = \mathrm{Proj}_{\mathcal{B}_p(\epsilon_p)}(\boldsymbol{\delta} + \eta\,\mathrm{sign}(\ell'(f_{\mathbf{w}}(\mathbf{x} + \boldsymbol{\delta}^i), y)))$ to approximately find a worst-case adversarial example, where $\ell'(f_{\mathbf{w}}(\mathbf{x} + \boldsymbol{\delta}^i), y)$ is the gradient of $\ell(f_{\mathbf{w}}(\mathbf{x} + \boldsymbol{\delta}^i), y)$ and $\mathrm{sign}(\cdot)$ is the sign function.

Tramer and Boneh [41] first proposed to solve the inner maximization problem of the problem (Equation (1)), by the MAX (the worst-case perturbation, Algorithm 1) and AVG (the augmentation of all perturbations, Algorithm 1). Now, the overall minimax objective becomes as below[2]

$$\text{MAX: } \min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)} \ell(f_{\mathbf{w}}(\mathbf{x} + \mathbf{MAX}(\mathbf{x})), y),$$

$$\text{AVG: } \min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)} \sum_{\boldsymbol{\delta} \in \mathbf{AVG}(\mathbf{x})} \ell(f_{\mathbf{w}}(\mathbf{x} + \boldsymbol{\delta}), y)\,.$$

[2]Here we omit most of the parameter of **MSD**, **AVG**, and **MAX** for the convenience of reading without compromising the important information.

Later, Maini *et al*. [26] designed a "greedy" algorithm named MSD, which solves the inner maximization problem by simultaneously maximizing the worst-case loss overall perturbation models at each projected steepest descent step as shown in Algorithm 1. And then the minimax objective becomes as

$$\text{MSD: } \min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)} \ell(f_{\mathbf{w}}(\mathbf{x} + \mathbf{MSD}(\mathbf{x})), y)\,.$$

3.2. Cooperative bargaining game

Cooperative bargaining game [40] is a process in which several parties jointly decide how to share a surplus that they can jointly gain. In the cooperative bargaining game, we have K players with their own utility function $u_i : \mathcal{A} \bigcup \{\mathbf{d}\} \to \mathbb{R}$, where $\mathcal{A}$ is the set of possible agreements and $\mathbf{d}$ is the disagreement point. The feasible set of utility is defined as $\mathcal{S} = \{(u_1(\boldsymbol{\gamma}) \ldots, u_K(\boldsymbol{\gamma})) : \boldsymbol{\gamma} \in \mathcal{A}\}$. The goals of players are to maximize their own utility functions. $\mathcal{S}$ is assumed to be convex and compact throughout this paper while there exists a point $\boldsymbol{\gamma} \in \mathcal{A}$ satisfying $u_i(\boldsymbol{\gamma}) > u_i(\mathbf{d}), \forall i \in [K]$ that strictly dominates the disagreement point $\mathbf{d}$, *i.e*., $u_i(\boldsymbol{\gamma}) > u_i(\mathbf{d}), \forall i \in [K]$, where $[K] = \{1, 2, \ldots, K\}$.

The multi-target adversarial training can be viewed as a cooperative game in which each target (perturbation) represents a player, whose utility is derived from the overall robust accuracy (defending ℓ_1, ℓ_2, and ℓ_∞ attacks simultaneously), and all the players negotiate to reach an agreed direction. We formalize the multi-target adversarial training problem as a bargaining game as follows. This bargaining game has K players and for each player, they generate a data-dependent perturbation $\boldsymbol{\delta}_k(\mathbf{x}), \forall k \in [K]$ to complete the adversarial training. The possible agreements $\mathcal{A}$ are $\{\sum_{k \in [K]} \boldsymbol{\gamma}_k = 1, \boldsymbol{\gamma}_k \geq 0, \forall k \in [K]\}$ and the disagreement points will be the set $\{\boldsymbol{\gamma}_k = 1, \boldsymbol{\gamma}_j = 0, \exists k \in [K], \forall j \in [K] \backslash \{k\}\}$, where $[K] \backslash \{k\}$ is the set containing integers from 1 to K without k. We note that the agreement set $\mathcal{A}$ is compact and convex. $\boldsymbol{\gamma}$ is used to aggregate the gradients and decide the final update direction. Specifically, for each updates (one data point, a mini-batch or an epoch) using gradient-based algorithms, the model is updated by $\mathbf{w} = \mathbf{w} - \eta \sum_{k \in [K]} \boldsymbol{\gamma}_k \ell'(f_{\mathbf{w}}(\mathbf{x} + \boldsymbol{\delta}_k), y)$, where η is the learning rate.

4. Convergence analysis

We begin this section by presenting our theoretical results based on the two commonly adopted machine learning models. Additionally, we have developed a general framework for multi-target adversarial training to avoid the player domination phenomenon that can cause the non-convergence of MAX and MSD in the next section. Our framework is inspired by our theoretical findings. All missing proofs are presented in Appendix A.

4.1. Convergence analysis on SVM model

Considering the binary classification setup [43], a data point $(\mathbf{x}, y)$ is sampled from a distribution $\mathcal{D}$ defined by

$$y \overset{\text{u.a.r}}{\sim} \{+1, -1\}, \quad \mathbf{x}_1 = \begin{cases} +y, & \text{w.p. } p; \\ -y, & \text{w.p. } 1-p, \end{cases},$$
$$\mathbf{x}_2, \ldots, \mathbf{x}_{d+1} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mu y, 1),$$

where $\mathbf{x} = [x_1, \ldots, x_{d+1}] \in \mathbb{R}^{d+1}$, y is a Rademacher random variable, and $\mathcal{N}(\mu, \sigma^2)$ is a normal distribution with mean μ and variance σ^2. In our setting, $p \in [0.5, 1]$. x_1 is a robust feature, while $\mathbf{x}_2, \ldots, \mathbf{x}_{d+1}$ are non-robust features that are weakly correlated with the label. Similarly, we set μ to be large enough such that a simple classifier can get a high standard accuracy ($> 99\%$), *i.e.*, $\mu \geq 1/\sqrt{d}$.

We train a linear model with soft SVM loss $\ell_{\text{soft}}(y', y) = \max(0, 1 - yy')$ on the data shown above

$$\min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \sum_{p\in\{1,2,\infty\}} \gamma_p \ell_{\text{soft}} \left(\mathbf{w}^\top(\mathbf{x} + \boldsymbol{\delta}_p), y\right), \quad \text{s.t. } \|\mathbf{w}\|_2 = 1, \tag{2}$$

where $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \gamma_\infty]$ satisfying $\sum_{i\in\{1,2,\infty\}} \gamma_i = 1$.

Let $\mathbf{w}^t$ and $\boldsymbol{\delta}^t$ be the weight vector and the perturbation at step t, respectively. The training procedures of the SVM model with AVG, MAX and MSD are illustrated as follows

0. Initialize the weights with natural training, *i.e.*, minimizing the soft-SVM loss without perturbation as

$$\mathbf{w}^0 = \operatorname{argmin}_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \ell_{\text{soft}} \left(\mathbf{w}^\top \mathbf{x}, y\right), \quad \text{s.t.} \quad \|\mathbf{w}\|_2 = 1. \tag{3}$$

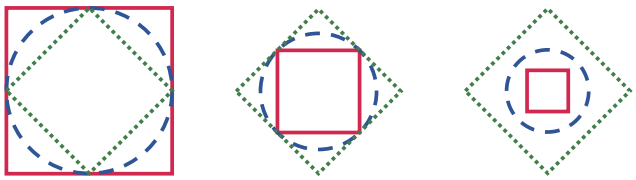
1. Get the optimal perturbations. With the linearity property of SVM, the closed form of optimal perturbations could be calculated by $\boldsymbol{\delta}^t_\infty = -y\epsilon_\infty \operatorname{sign}(\mathbf{w}^t)$, $\boldsymbol{\delta}^t_1 = \frac{-y\epsilon_1 \mathbf{w}^t}{\|\mathbf{w}^t\|_1}$, $\boldsymbol{\delta}^t_2 = \frac{-y\epsilon_2 \mathbf{w}^t}{\|\mathbf{w}^t\|_2}$. at time t.

2. Update the weights $\mathbf{w}^t$ with MAX, MSD, or AVG by

$$\mathbf{w}^t = \operatorname{argmin}_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \sum_{p\in\{1,2,\infty\}} \gamma^t_p \cdot \ell_{\text{soft}} \left(\mathbf{w}^\top(\mathbf{x} + \boldsymbol{\delta}^t_p), y\right), \quad \text{s.t. } \|\mathbf{w}^t\|_2 = 1,$$

where $\boldsymbol{\gamma}^t = [1/3, 1/3, 1/3]$ if the algorithm is AVG; $\boldsymbol{\gamma}^t \in \{[1,0,0], [0,1,0], [0,0,1]\}$ if the algorithm is MAX or MSD.

3. Loop Steps 1 and 2 for predefined number of epochs or until convergence.



(a) $d = 2$ (b) $d = 4$ (c) $d = 8$

Figure 2. Illustration of feasible domains of ℓ_∞- (red region), ℓ_1- (green dotted region), and ℓ_2- (blue dashed region) players in $\mathbb{R}^2$, when the budgets satisfy the minimum requirements of Theorem 1, *i.e.*, $\epsilon_\infty = \frac{2}{d}\epsilon_1 = \sqrt{\frac{2}{d}}\epsilon_2$. We notice that when $d = 2$, the feasible regions of 1- and 2-players are contained in the region of ℓ_∞-player, while with the increase of dimension of data, the inverse case occurs and the feasible region of ℓ_∞-player is strictly dominated by that of ℓ_1-player. Best view in color.

We first present the following negative result,

Theorem 1. *Let $\mu \geq 4/\sqrt{d}$, $\epsilon_\infty \geq 2\mu$, $p \leq 0.977$. If one uses MAX and MSD to train the soft SVM model given $\epsilon_\infty \geq \frac{2}{d}\epsilon_1$ and $\epsilon_\infty \geq \sqrt{\frac{2}{d}}\epsilon_2$, the loss incurred by the ℓ_∞-player (ℓ_∞-adversary) is larger than that by the ℓ_1-player (ℓ_1-adversary) and the ℓ_2-player (ℓ_2-adversary) at any time t for any data sampled from the distribution $\mathcal{D}$,* i.e., *$\ell_{soft}\left(\mathbf{w}^\top(\mathbf{x} + \boldsymbol{\delta}^t_\infty), y\right) \geq \max_{p\in\{1,2\}} \ell_{soft}\left(\mathbf{w}^\top(\mathbf{x} + \boldsymbol{\delta}^t_p), y\right), \forall t, \forall(\mathbf{x}, y) \sim \mathcal{D}$. Furthermore, $\boldsymbol{\gamma}_1 = \boldsymbol{\gamma}_2 = 0$ and $\boldsymbol{\gamma}_\infty = 1$ with MAX and MSD, which means the training dynamics of SVM model with MAX and MSD are controlled by the ℓ_∞-player.*

Remark 1. *This theorem shows that even when the feasible domain of ℓ_∞-adversary is much smaller than that of ℓ_1- and ℓ_2- adversaries (when the dimension d of data is bigger than 2), the training dynamics of SVM will still be controlled by the ℓ_∞-player. By the definition of bargaining game in multi-target adversarial training, at any time t, the models update with the disagreement points. As shown in Figure 2, with the increase of dimension of the data, the feasible domain of ℓ_∞-adversary is strictly contained in the ℓ_1-players' region.*

We define the phenomenon where one player "dominates" the multi-target adversarial training procedure (the training procedure only depends on one player) as follows

Definition 2 (Player dominates the cooperative game)**.** *If $\exists i \in [k]$ such that $\boldsymbol{\gamma}^t_i = 1$ and $\boldsymbol{\gamma}^t_j = 0, \forall j \in [K]/\{i\}, \forall t$, then we call that i-player dominates the bargaining game as models achieve the same disagreement point at any time t.*

Further, we observe that this phenomenon might lead to the **non-convergence** of SVM with MAX and MSD as the sign of weights of the model flips over time when ℓ_∞-player dominates the bargaining game, and given $\epsilon_\infty > \mu$.

Theorem 3. *Consider Problem (Equation (2)) trained with MAX and MSD. If ℓ_∞-player dominates the bargaining*

game (player domination) and $\epsilon_\infty > \mu$, the weights for the non-robust features flip over time, i.e., $\text{sign}(\mathbf{w}_i^t) = -\text{sign}(\mathbf{w}_i^{t-1}), \forall i \geq 2, \forall t$. *Thus, the training procedure with MAX and MSD does not converge.*

Although we only analyze the case when the ℓ_∞-player dominates the bargaining game, we notice that in situations where other players dominate this bargaining game (also known as multi-target adversarial training), with certain conditions such as $\epsilon_1 > 2\mu$, the training procedure may not converge empirically. Motivated by the negative results of the SVM model, we next test a conjecture that player domination may also lead to non-convergence in linear models.

4.2. Player domination leads to non-convergence

To test our conjecture, we introduce a linear model as follows. The linear model $f_{\mathbf{w}}$ is parameterized by $\mathbf{w}$ and optimized by gradient-based algorithms such as AdaGrad [10] or Adam [20]. The parameter at time (epoch) t is denoted by $\mathbf{w}^t$. The loss function of each player is denoted by ℓ_k, where $k \in [K]$, which is L-smooth and μ-strongly convex, and the corresponding gradient at time t is denoted as g_k^t, where $k \in [K]$ for all t. We assume that for a sequence $\{\mathbf{w}^t\}_{t\in[1,\infty]}$ generated by any gradient-based optimization algorithm, the set of gradient vectors $\{g_k^t\}_{k\in[K]}$ at any time t and at any partial limit is linearly independent unless a locally optimal solution is achieved. All loss functions are differentiable, and all sub-level sets are bounded. The learning rate is denoted by η such that $\eta < \frac{2}{L}$. We also assume that the domain of weights is open and convex.

To generalize our theoretical results, we show that under this linear model, MAX and MSD still do not converge if one player dominates the game.

Theorem 4. *Consider using MAX and MSD to train the linear model described above. If one player dominates the bargaining game throughout the game (see Definition 2), then the loss of all players and the overall loss would increase as time t grows. This means that the training procedure for the linear model described above does not converge.*

While we have shown that MAX and MSD do not converge under the two models that we study, we notice that AVG provably converges as the loss is decreasing *w.r.t* the number of epochs. See the following theorem.

Theorem 5. *Using AVG to train the linear model, the overall loss decreases as time t grows.*

This theorem shows that under the same setting, while the loss of each player and the overall loss will increase as time grows with MAX and MSD, the overall loss will decrease with AVG. The key factor that results in the non-convergence phenomenon with MAX and MSD is the player domination phenomenon, where players reach the same disagreement point all the time, leading to an increase in loss. Since AVG does not achieve any disagreement point, the player domination phenomenon does not occur, and convergence is possible. Therefore, the key to avoiding the non-convergence of MAX and MSD may be to avoid player domination, which inspires us to design the new algorithm introduced in the next section.

Algorithm 2 Framework of Multi-target Adversarial Training with **AdaptiveBudget**

Require: Training epochs E, dataset $(\mathcal{X}, \mathcal{Y})$, adversarial budgets $(\epsilon_\infty, \epsilon_1, \epsilon_2)$, model $f(\cdot)$, loss function ℓ.

for $e \in [E]$ **do**
  for $\mathbf{x}, y \in (\mathcal{X}, \mathcal{Y})$ **do**
    $\boldsymbol{\delta}_p(\mathbf{x}) \leftarrow \text{PGD}(\mathbf{x}, k, \eta, \epsilon_p, \ell, f), g_p \leftarrow \ell'(f(\mathbf{x}+\boldsymbol{\delta}_p(\mathbf{x})), y), \forall p \in \{1, 2, \infty\}$;
    Get adaptive budgets $\hat{\epsilon}_1, \hat{\epsilon}_2, \hat{\epsilon}_\infty \leftarrow$ **AdaptiveBudget** $([g_1, g_2, g_\infty], [\epsilon_1, \epsilon_2, \epsilon_\infty])$;
    Adversarial training using MAX, MSD or AVG with budgets $(\hat{\epsilon}_1, \hat{\epsilon}_2, \hat{\epsilon}_\infty)$;
  end for
end for
Return the classifier f.

AdaptiveBudget$([g_1, g_2, g_\infty], [\epsilon_1, \epsilon_2, \epsilon_\infty])$:

$p_{\text{max}} \leftarrow \text{argmax}_{p\in\{\infty,1,2\}} \|g_p\|$;
$p_{\text{min}} \leftarrow \text{argmin}_{p\in\{\infty,1,2\}} \|g_p\|$;
$p_{\text{mid}} \leftarrow \{1, 2, \infty\} / \{p_{\text{max}}, p_{\text{min}}\}$;
$\epsilon_{p_{\text{max}}} \leftarrow \epsilon_{p_{\text{max}}} \cdot \frac{\|g_{p_{\text{max}}}\|}{\|g_{p_{\text{mid}}}\|}, \quad \epsilon_{p_{\text{min}}} \leftarrow \epsilon_{p_{\text{min}}} \cdot \frac{\|g_{p_{\text{min}}}\|}{\|g_{p_{\text{mid}}}\|}$;
Return $\epsilon_1, \epsilon_2, \epsilon_\infty$.

5. Avoiding player domination via AdaptiveBudget

In this section, we present the proposed algorithm **AdaptiveBudget** summarized in Algorithm 2.

Our theoretical results (Theorem 3 and Theorem 4) show that MAX and MSD cannot converge when player domination occurs (Definition 2). Indeed, to achieve convergence of the model, researchers can directly use AVG [41] instead of MAX [41] and MSD [26]. However, previous works [26, 41] have shown that under the non-convex scenario, where a deep neural network with non-linear activation is trained on MNIST [23] and CIFAR-10 [21], MSD and MAX outperform AVG[3]. We have also come to a similar conclusion as shown in Table 1 and Table 2. Therefore, inspired by the previous theoretical analysis, to avoid player domination, we increase the budget of the player with the largest gradient

[3]This does not conflict with our theoretical analysis as the training dynamics of non-convex and convex scenarios (*e.g.*, SVM and linear models) are different. Additionally, since MAX [41] and MSD [26] are greedy algorithms that take steepest gradients at each time t, such greedy updates benefit under non-convex scenarios.

and force the model to better handle this adversary. Intuitively, if the model can handle one adversary (player) well, the gradient of that adversary (player) will be small. So, to advance multi-target robustness, we present a novel general-purpose algorithm for multi-target adversarial robustness called **AdaptiveBudget**, which adaptively changes the budget of different adversaries to avoid the player domination phenomenon (achieving the same disagreement point).

The core idea of this algorithm is to avoid player domination by adaptively assigning proper attack budgets to different adversaries (players). Such an assignment is intended to ensure that no single player's loss is significantly larger than others, and thus alleviate player domination. In each epoch, the player who controls the updates will be different. Concretely, for each batch of data, we first obtain adversarial perturbations $\boldsymbol{\delta}_\infty$, $\boldsymbol{\delta}_1$, and $\boldsymbol{\delta}_2$ for the ℓ_∞-, ℓ_1-, and ℓ_2- adversaries (Step 4). Then, based on the norms (ℓ_1 or ℓ_2 norms) of the gradients by forwarding their adversarial examples through our model, the algorithm adaptively adjusts the budgets ϵ for different adversaries to avoid the player domination phenomenon (Step 5). Specifically, our proposed method does not change the budget of the adversary whose norm of gradient is the middle one, increases the budget of the adversary whose norm of gradient is the maximum, and decreases the budget of the adversary whose norm of gradient is the minimum. The intuition behind our method is to focus on the hardest task in the current round so that this task might be easier to model in the next round and might not be able to dominate the updates. After obtaining the adjusted adversarial budgets, the model utilizes MSD, MAX, or AVG to approximately solve the inner maximization problem and then updates its parameter with a gradient descent algorithm.

The proposed framework is general and can be applied to all existing multi-target adversarial training algorithms. The **AdaptiveBudget** module is employed to break the curse of player domination, which might occur when applying MAX and MSD to train a robust model. In the next section, we provide extensive experimental evidence to support the consistent effectiveness of the AdaptiveBudget method.

6. Experiments

6.1. Experimental setup and implementation details

Datasets. We conducted extensive experiments on one synthetic dataset (Sec. 4.1) to complement our theoretical results, and on MNIST [23], CIFAR-10 [21], and CIFAR-100 [21] to show the superiority of our proposed methods over the existing methods of multi-target adversarial training. Due to the limitation of space, the experiments on synthetic data is in Appendix.

Methods. Models that defend against multiple adversaries are trained using MAX [41], AVG [41], and MSD [26]. For each algorithm, we use the default hyperparameters introduced in their original papers. All methods are implemented in PyTorch [32] on a single NVIDIA A100 GPU. Raw images are resized to 28×28 pixels for MNIST and 32×32 pixels for CIFAR-10 and CIFAR-100 as inputs. We apply the AdaptiveBudget to MAX, MSD, and AVG with ℓ_1 and ℓ_2 norms to assign proper budgets adaptively to avoid player domination.

Models. Following MSD [26] and Madry *et al*. [25], for MNIST, we use a four-layer convolutional network which consists of two convolutional layers of 32 and 64 5×5 filters and 2 units of padding, followed by a fully connected layer with 1024 hidden units, where both convolutional layers are followed by 2×2 Max Pooling layers and ReLU activations. Similarly, following MSD [26], for CIFAR-10 and CIFAR-100, we use the pre-activation version of the ResNet18 [16] architecture that consists of nine residual units with two convolutional layers.

Attacks used for training. For MNIST, we follow the setting of three adversaries from MSD [26], as shown below. The ℓ_∞-adversary uses a step size of $\alpha = 0.01$ within a radius of $\epsilon_\infty = 0.3$ for 50 iterations. The ℓ_2-adversary uses a step size of $\alpha = 0.1$ within a radius of $\epsilon_2 = 2.0$ for 100 iterations, and the ℓ_1-adversary uses a step size of $\alpha = 0.8$ within a radius of $\epsilon_1 = 10$ for 50 iterations. By default, the attack is run with two restarts: one starting with $\boldsymbol{\delta} = \mathbf{0}$, and another by randomly initializing $\boldsymbol{\delta}$ in the perturbation ball. Similarly, for CIFAR-10 and CIFAR-100, we follow MSD [26]. The ℓ_∞-adversary uses a step size of $\alpha = 0.003$ within a radius of $\epsilon_\infty = 0.03$ for 40 iterations. The ℓ_2-adversary uses a step size of $\alpha = 0.05$ within a radius of $\epsilon_2 = 0.5$ for 50 iterations, and the ℓ_1-adversary uses a step size of $\alpha = 1.0$ within a radius of $\epsilon_1 = 12$ for 50 iterations.

Attacks used for evaluation. To fully understand the performance of the defense, we employ the PGD adversary and Autoattack [8][4] to test the effectiveness of our method. We make 10 random restarts for all results on MNIST, CIFAR-10, and CIFAR-100. The budgets for the three adversaries, *i.e*., ϵ_1, ϵ_2, and ϵ_∞, are the same as the setting during training for both datasets. However, we increase the number of iterations to $(100, 200, 100)$ for $(\ell_\infty, \ell_2, \ell_1)$ on MNIST, and to $(100, 500, 100)$ for $(\ell_\infty, \ell_2, \ell_1)$ on CIFAR-10 and CIFAR-100.

Hyperparameter setting and tuning. We did not tune any hyperparameters as our goal is to demonstrate the player domination phenomenon and propose a solution with our **AdaptiveBudget** method. We adopted all hyperparameters directly from MSD [26]. Specifically, on MNIST, we used Adam [20] without weight decay and a variation of the learning rate schedule from Smith [38]. The schedule is piecewise

[4] We only consider white-box attacks based on gradients and do not use attacks based on gradient estimation, as the gradients for the standard architectures used here are readily available.

Table 1. Summary of robust accuracy for MNIST (higher is better). "w. AdaptiveBudget" refers to employing AdaptiveBudget which aims to avoid any player dominating the game. "*" means that the results are reproduced from the implementation of MSD [26] with the hyperparameters introduced in MSD [26]. "ℓ_1 (ours)" and "ℓ_2 (ours)" refers to employing our proposed AdaptiveBudget method w.r.t ℓ_1 and ℓ_2 norms. Note that multi-target robustness focuses on the **overall robust accuracy** ("**All Robust Acc**" in the table).

Models w. **AdaptiveBudget**	ℓ_1	ℓ_2	ℓ_∞	MAX	MAX ℓ_1 (ours)	MAX ℓ_2 (ours)	MSD	MSD ℓ_1 (ours)	MSD ℓ_2 (ours)	AVG	AVG ℓ_1 (ours)	AVG ℓ_2 (ours)
Clean Accuracy (%)	97.2*	99.1*	99.2*	98.6*	**98.9**↑	**98.9**↑	98.2*	**98.3**↑	**98.9**↑	99.1*	99.1	99.1
ℓ_1 PGD Robust Acc (%)	47.3*	67.8*	54.6*	67.1*	**71.4**↑	**69.7**↑	67.3*	66.8↓	65.9↓	70.6*	68.2↓	68.9↓
ℓ_2 PGD Robust Acc (%)	24.1*	66.8*	61.8*	67.2*	**69.4**↑	**69.5**↑	68.0*	67.9↓	65.3↓	69.4*	68.3↓	68.3↓
ℓ_∞ PGD Robust Acc (%)	0*	0.1*	88.9*	21.2*	**67.2**↑	**67.6**↑	62.4*	**69.7**↑	**69.7**↑	59.5*	**67.7**↑	**65.6**↑
All PGD Robust Acc (%)	0*	0.1*	52.1*	21.2*	**61.3**↑	**61.4**↑	59.7*	**62.1**↑	**61.0**↑	55.4*	**59.2**↑	**58.2**↑

linear, starting from 0 and increasing to 10^{-3} over the first 6 epochs, then decreasing to 0 over the last 9 epochs. On CIFAR-10 and CIFAR-100, we used SGD [34] with momentum 0.9 and weight decay 5×10^{-4} for all models. We also used a variation of the learning rate schedule from Smith [38] to achieve superconvergence in 50 epochs. The schedule is piecewise linear, starting from 0 and increasing to 0.1 over the first 20 epochs, then decreasing to 0.005 over the next 20 epochs, and finally decreasing to 0 over the last 10 epochs.

Evaluation metric. While our main target is to improve the **overall robust accuracy** on ℓ_1-, ℓ_2, and ℓ_∞- attacks, we report the single attack accuracy as well. The overall robust accuracy is calculated as $\sum_{(\mathbf{x},y)}(\mathbf{I}(f(\mathbf{x}+\boldsymbol{\delta}_1(\mathbf{x})) == y) * \mathbf{I}(f(\mathbf{x}+\boldsymbol{\delta}_2(\mathbf{x})) == y) * \mathbf{I}(f(\mathbf{x}+\boldsymbol{\delta}_\infty(\mathbf{x})) == y))/n$, where $\mathbf{I}(cond) = 1$ when $cond$ is true and $\mathbf{I}(cond) = 0$ when $cond$ is false, n is the total number of testing data, and $f(\cdot)$ is the trained model.

6.2. Results on MNIST

Here we present results on the MNIST dataset, summarized in Table 1. Although it has been considered as an "easy" benchmark compared to CIFAR-10 or larger datasets, such as ImageNet [9], we noticed that all the single target adversarial training methods, namely ℓ_1, ℓ_2, and ℓ_∞, fail to defend against only three attacks, while the best method is ℓ_∞ training, which defends against almost all three attacks and outperforms the MAX method.

From Table 1, we can see that our proposed AdaptiveBudget improves the overall robust accuracy against ℓ_1, ℓ_2, and ℓ_∞ PGD attacks, as well as the ℓ_∞ robust accuracy for all three methods, *i.e.*, MAX, MSD, and AVG, using both ℓ_1 and ℓ_2 norms. Specifically, on MAX, the ℓ_1 and ℓ_2 robust accuracy is improved by 4.3% and 2.2% (with ℓ_1 norm AdaptiveBudget), 2.6% and 2.3% (with ℓ_2 norm AdaptiveBudget), respectively. Additionally, we observe that our proposed method is able to avoid the player domination phenomenon even in non-convex scenarios, as it improves the all PGD robust accuracy of MAX by over 40%, and all the robust accuracies of MAX are improved.

The all PGD robust accuracy of vanilla MAX also shows that the player domination phenomenon hinders MAX from achieving satisfactory robust accuracy for non-convex scenarios. Maini *et al*. [26] and Tramer and Boneh [41] mention that there is a trade-off between robust accuracy against ℓ_∞ attacks and robust accuracy against ℓ_1 and ℓ_2 attacks. Similar observations can be obtained from our experimental results. For MSD and AVG, the robust accuracy defending ℓ_1 and ℓ_2 PGD attacks drops slightly, which might be due to this trade-off.

Norm choice in AdaptiveBudget. We use ℓ_1 and ℓ_2 norms for AdaptiveBudget, and the corresponding results are shown in Table 1. There is no significant difference between the experiments with ℓ_1 and ℓ_2 norms when using our proposed method. The differences in overall robust accuracy are only 0.1%, 1.1%, and 1.0% on MAX, MSD, and AVG, respectively. The differences in separated robust accuracy are also small, which proves the generalization ability of our proposed method empirically.

6.3. Results on CIFAR-10 and CIFAR-100

The results are shown in Tables 2 and 3, and the curve of robust accuracy on CIFAR-10 is shown in Figure 6 in the Appendix. Due to the limitation of space, we present the most important results in the main paper while leaving the left results in the Appendix.

Main results. The results on CIFAR-10 presented in Table 2 show the generalization ability of our proposed method, which improves the overall robust accuracy of PGD and AutoAttack of three methods, *i.e.*, MSD, MAX, and AVG. We notice that the overall robust accuracy for PGD and AutoAttack is mainly restricted by how well the model defends against the ℓ_∞ attack. This might be caused by the fact that the radius of the ℓ_∞ attack is too small compared to the radius of the ℓ_1 and ℓ_2 attacks, so with the updates by gradient-based algorithms, the gradient of the ℓ_∞ adversary is covered by the others, causing the model to ignore the ℓ_∞ adversary. Furthermore, we notice that employing AdaptiveBudget with either the ℓ_1 or ℓ_2 norms helps models pay attention to the tasks that are not well-learned as the ℓ_∞ robust accuracy is relatively improved the most. For example, the ℓ_∞ PGD robust accuracy of MAX with AdaptiveBudget

Table 2. Summary of robust accuracy for CIFAR-10 (higher is better). "w. AdaptiveBudget" refers to employing AdaptiveBudget which aims to avoid any player dominating the game. "AA" refers to AutoAttack. "*" means that the results are reproduced from the implementation of MSD [26] with the hyperparameters introduced in MSD [26]. "ℓ_1 (ours)" and "ℓ_2 (ours)" refers to employing our proposed AdaptiveBudget method w.r.t ℓ_1 and ℓ_2 norms. Note that multi-target robustness focuses on the **overall robust accuracy** ("**All Robust Acc**" in the table).

Models w. **AdaptiveBudget**	ℓ_1	ℓ_2	ℓ_∞	MAX	MAX ℓ_1 (ours)	MAX ℓ_2 (ours)	MSD	MSD ℓ_1 (ours)	MSD ℓ_2 (ours)	AVG	AVG ℓ_1 (ours)	AVG ℓ_2 (ours)
Clean Accuracy	92.4*	87.5*	84.2*	79.6*	76.9	78.7	79.2*	77.6	79.0	83.8*	81.6	81.5
ℓ_1 PGD Robust Acc (%)	90.8*	31.7*	17.3*	44.0*	**50.7**↑	**51.7**↑	50.8*	**51.2**↑	**52.6**↑	55.7*	**57.3**↑	**56.3**↑
ℓ_2 PGD Robust Acc (%)	0.1*	64.0*	60.6*	55.6*	**63.4**↑	**65.1**↑	64.3*	63.6↓	**65.5**↑	67.0*	66.6↓	67.0
ℓ_∞ PGD Robust Acc (%)	0*	27.8*	51.2*	41.3*	**47.5**↑	**47.6**↑	45.7*	**48.4**↑	**47.2**↑	39.4*	**45.5**↑	**44.2**↑
All PGD Robust Acc (%)	0*	23.8*	17.3*	40.4*	**46.0**↑	**46.8**↑	44.1*	**47.2**↑	**46.4**↑	39.2*	**45.2**↑	**43.6**↑
ℓ_1 AA Robust Acc (%)	0*	23.8*	6.2*	41.4*	**45.7**↑	**45.5**↑	45.5*	**46.4**↑	**46.7**↑	49.7*	**52.7**↑	**50.8**↑
ℓ_2 AA Robust Acc (%)	0*	63.0*	57.4*	53.7*	**60.4**↑	**63.2**↑	61.9*	**62.3**↑	**62.1**↑	65.4*	64.6↓	**65.5**↑
ℓ_∞ AA Robust Acc (%)	0*	26.1*	48.0*	38.4*	**44.7**↑	**44.1**↑	43.1*	**45.2**↑	**44.4**↑	37.0*	**43.1**↑	**42.1**↑
All AA Robust Acc (%)	0*	19.5*	6.2*	37.6*	**42.9**↑	**42.3**↑	41.6*	**43.4**↑	**43.0**↑	36.6*	**42.5**↑	**41.2**↑

Table 3. Summary of robust accuracy for CIFAR-100 (higher is better). "w. AdaptiveBudget" refers to employing AdaptiveBudget which aims to avoid any player dominating the game. "AA" refers to AutoAttack. "*" means that the results are reproduced from the implementation of MSD [26] with the hyperparameters introduced in MSD [26]. "ℓ_1 (ours)" and "ℓ_2 (ours)" refers to employing our proposed AdaptiveBudget method w.r.t ℓ_1 and ℓ_2 norms.

Models w. **AdaptiveBudget**	MAX	MAX ℓ_1 (ours)	MAX ℓ_2 (ours)	MSD	MSD ℓ_1 (ours)	MSD ℓ_2 (ours)	AVG	AVG ℓ_1 (ours)	AVG ℓ_2 (ours)
Clean Accuracy	55.49*	56.48	55.53	56.09*	55.52	54.94	59.94*	57.78	58.16
ℓ_1 PGD Robust Acc (%)	25.45*	**29.27**↑	**29.78**↑	35.50*	30.31↓	28.87↓	30.35*	**33.16**↑	**32.62**↑
ℓ_2 PGD Robust Acc (%)	39.55*	**40.00**↑	**39.85**↑	40.14*	**40.28**↑	39.28↓	40.26*	**41.03**↑	**40.27**↑
ℓ_∞ PGD Robust Acc (%)	25.03*	**25.34**↑	**25.87**↑	24.83*	**26.19**↑	**25.59**↑	18.92*	**21.81**↑	**21.57**↑
All PGD Robust Acc (%)	21.11*	**24.14**↑	**24.76**↑	25.10*	25.03↓	24.43↓	18.61*	**21.55**↑	**21.16**↑
ℓ_1 AA Robust Acc (%)	13.00*	**23.00**↑	**20.90**↑	25.10*	24.00↓	24.20↓	25.20*	**28.60**↑	**28.00**↑
ℓ_2 AA Robust Acc (%)	36.30*	35.60↓	**36.40**↑	37.60*	35.80↓	36.40↓	37.00*	**37.90**↑	**37.10**↑
ℓ_∞ AA Robust Acc (%)	22.00*	21.50↓	**22.30**↑	21.80*	**22.80**↑	**22.70**↑	16.30*	**19.00**↑	**19.70**↑
All AA Robust Acc (%)	12.20*	**20.60**↑	**18.60**↑	21.00*	**21.30**↑	**21.50**↑	16.10*	**18.90**↑	**19.50**↑

w.r.t. the ℓ_1 norm experiences a relative 15.01% improvement, while there is only a 14.03% relative improvement on the ℓ_2 PGD robust accuracy. In addition, the trade-off between the three attacks on CIFAR-10 is different from that on MNIST. On MNIST, the ℓ_2 robust accuracy is related to that of the ℓ_1 adversary, while on CIFAR-10, it seems that ℓ_2 robust accuracy is more likely to be related to ℓ_∞ robust accuracy. Similar observations can be obtained on CIFAR-100 in Table 3.

7. Conclusion

In this paper, to achieve the ultimate goal of robustness, *i.e.*, defending any terms of attacks, we first formalized this problem within the context of a bargaining game and investigated the convergence properties of MAX, MSD, and AVG under two machine learning cases. We discovered that MAX and MSD do not converge theoretically due to a phenomenon called player domination, while AVG does not suffer from this. To prevent player domination during the training of robust models, we designed a novel framework for multi-target adversary training, which includes the proposed AdaptiveBudget method. Specifically, AdaptiveBudget adaptively changed the budget of different attacks to avoid player domination based on the norm of gradients of each adversary. Finally, to evaluate the proposed framework, we conducted experiments on three benchmarks, *i.e.*, MNIST, CIFAR-10, and CIFAR-100. Experimental results showed that AdaptiveBudget improved the overall robust accuracy on three benchmarks, which complemented our theoretical results and also supported our finding that player domination might interfere with the training of robust models.

Ethics Statement

Our work strongly relates to the security of machine learning, especially defending against adversarial examples. Numerous studies [3, 12, 17, 22, 36] have shown that adversarial examples can cause modern machine learning systems to fail. To improve the robustness of machine learning models, both empirically and theoretically, various approaches [2, 6, 42, 45, 58] have been proposed. While the ultimate goal of robust machine learning is to defend against all possible and reasonable adversarial examples, most of the previous methods have only been proven to defend against one type of attack [18, 26]. To move towards multi-target robustness, previous studies [26, 41] have proposed methods that ensure robustness towards ℓ_1, ℓ_2, and ℓ_∞-adversaries simultaneously. However, these methods are mainly empirical and have not been thoroughly explored theoretically. To address this gap, we investigated this problem theoretically and proposed a method that can improve the performance of all the previous methods. We believe that improving the overall worst-case robustness of machine learning models may lead to the ultimate goal of robustness, which is to learn a model that can defend against all types of attacks.

Acknowledgement

This work is supported by NSERC Discovery Grant RGPIN-2022-03215, DGECR-2022-00357.

References

[1] 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*, pages 274–283, 2018. 2

[2] Mislav Balunovic and Martin Vechev. Adversarial training and provable defenses: Bridging the gap. In *International Conference on Learning Representations*, 2020. 9

[3] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations*, 2018. 9

[4] Wieland Brendel, Jonas Rauber, Matthias Kümmerer, Ivan Ustyuzhaninov, and Matthias Bethge. Accurate, reliable and fast robustness evaluation. In *Advances in Neural Information Processing Systems*, pages 12841–12851, 2019. 2

[5] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. In *IEEE symposium on security and privacy*, pages 39–57, 2017. 2

[6] Pin-Chun Chen, Bo-Han Kung, and Jun-Cheng Chen. Class-aware robust adversarial training for object detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10420–10429, 2021. 9

[7] Francesco Croce and Matthias Hein. Provable robustness against all adversarial l_p-perturbations for $p \geq 1$. In *International Conference on Learning Representations*, 2020. 1, 2

[8] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning*, pages 2206–2216, 2020. 2, 6

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition*, pages 248–255, 2009. 7

[10] John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011. 3, 5

[11] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A Rotation and a Translation Suffice: Fooling CNNs with Simple Transformations. 2018. 1

[12] Shuo Feng, Xintao Yan, Haowei Sun, Yiheng Feng, and Henry X Liu. Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment. *Nature communications*, 12(1):1–14, 2021. 9

[13] Ruize Gao, Jiongxiao Wang, Kaiwen Zhou, Feng Liu, Binghui Xie, Gang Niu, Bo Han, and James Cheng. Fast and reliable evaluation of adversarial robustness with minimum-margin attack. In *International Conference on Machine Learning*, pages 7144–7163, 2022. 2

[14] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *International Conference on Learning Representations*, 2015. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 1

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 6

[17] Yunhan Jia, Yantao Lu, Junjie Shen, Qi Alfred Chen, Hao Chen, Zhenyu Zhong, and Tao Wei. Fooling de-

tection alone is not enough: Adversarial attack against multiple object tracking. In *International Conference on Learning Representations*, 2020. 9

[18] Daniel Kang, Yi Sun, Tom Brown, Dan Hendrycks, and Jacob Steinhardt. Transfer of Adversarial Robustness Between Perturbation Types. In *arxiv:1905.01034*, 2019. 1, 2, 9

[19] Harini Kannan, Alexey Kurakin, and Ian Goodfellow. Adversarial Logit Pairing. In *arXiv:1803.06373*, 2018. 1, 2

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 3, 5, 6

[21] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009. 5, 6

[22] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *International Conference on Learning Representations Workshop*, 2017. 2, 9

[23] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, and others. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5, 6

[24] Yandong Li, Lijun Li, Liqiang Wang, Tong Zhang, and Boqing Gong. Nattack: Learning the distributions of adversarial examples for an improved black-box attack on deep neural networks. In *International Conference on Machine Learning*, pages 3866–3876, 2019. 2

[25] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. 1, 2, 6

[26] Pratyush Maini, Eric Wong, and J. Zico Kolter. Adversarial Robustness Against the Union of Multiple Perturbation Models. In *International Conference on Machine Learning*, pages 6640–6650, 2020. 1, 2, 3, 5, 6, 7, 8, 9

[27] Marius Mosbach, Maksym Andriushchenko, Thomas Trost, Matthias Hein, and Dietrich Klakow. Logit pairing methods can fool gradient-based attacks. In *NeurIPS 2018 Workshop on Security in Machine Learning*, 2018. 2

[28] Tianyu Pang, Kun Xu, and Jun Zhu. Mixup inference: Better exploiting mixup to defend adversarial attacks. In *International Conference on Learning Representations*, 2020. 2

[29] Tianyu Pang, Xiao Yang, Yinpeng Dong, Hang Su, and Jun Zhu. Bag of tricks for adversarial training. In *International Conference on Learning Representations*, 2021. 2

[30] Tianyu Pang, Xiao Yang, Yinpeng Dong, Kun Xu, Jun Zhu, and Hang Su. Boosting adversarial training with hypersphere embedding. In *Advances in Neural Information Processing Systems*, pages 7779–7792, 2020. 2

[31] Nicolas Papernot, Patrick D. 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*, pages 582–597, 2016. 1, 2

[32] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019. 6

[33] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified Defenses against Adversarial Examples. In *International Conference on Learning Representations*, 2022. 1

[34] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951. 7

[35] Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first adversarially robust neural network model on MNIST. In *International conference on Learning Representations*, 2019. 2

[36] Ali Shafahi, W. Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. Are adversarial examples inevitable? In *International Conference on Learning Representations*, 2019. 9

[37] Baifeng Shi, Dinghuai Zhang, Qi Dai, Zhanxing Zhu, Yadong Mu, and Jingdong Wang. Informative dropout for robust representation learning: A shape-bias perspective. pages 8828–8839, 2020. 2

[38] Leslie N. Smith. A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay. In *arXiv:1803.09820*, 2018. 6, 7

[39] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. 1

[40] William Thomson. Chapter 35 Cooperative models of bargaining. In *Handbook of Game Theory with Economic Applications*, volume 2, pages 1237–1284. Elsevier, 1994. 2, 3

[41] Florian Tramer and Dan Boneh. Adversarial Training and Robustness for Multiple Perturbations. In *Advances in Neural Information Processing Systems*, pages 5858–5868, 2019. 2, 3, 5, 6, 7, 9

[42] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations*, 2018. 9

[43] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness May Be at Odds with Accuracy. In *International Conference on Learning Representations*, 2019. 4, 12

[44] Haotao Wang, Tianlong Chen, Shupeng Gui, Ting-Kuei Hu, Ji Liu, and Zhangyang Wang. Once-for-all adversarial training: In-situ tradeoff between robustness and accuracy for free. In *Advances in Neural Information Processing Systems*, 2020. 2

[45] Hongjun Wang and Yisen Wang. Self-ensemble adversarial training for improved robustness. In *International Conference on Learning Representations*, 2022. 9

[46] Qizhou Wang, Feng Liu, Bo Han, Tongliang Liu, Chen Gong, Gang Niu, Mingyuan Zhou, and Masashi Sugiyama. Probabilistic margins for instance reweighting in adversarial training. In *Advances in Neural Information Processing Systems*, pages 23258–23269, 2021. 2

[47] Yimu Wang, Shiyin Lu, and Lijun Zhang. Searching privately by imperceptible lying: A novel private hashing method with differential privacy. In *ACM International Conference on Multimedia*, page 2700–2709, 2020. 1

[48] Yimu Wang, Ren-Jie Song, Xiu-Shen Wei, and Lijun Zhang. An adversarial domain adaptation network for cross-domain fine-grained recognition. In *IEEE Winter Conference on Applications of Computer Vision*, pages 1217–1225, 2020. 1

[49] Eric Wong and Zico Kolter. Provable Defenses against Adversarial Examples via the Convex Outer Adversarial Polytope. In *International Conference on Machine Learning*, pages 5286–5295, 2018. 1

[50] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. In *Advances in Neural Information Processing Systems*, pages 2958–2969, 2020. 2

[51] Yihan Wu, Aleksandar Bojchevski, and Heng Huang. Adversarial weight perturbation improves generalization in graph neural network. *arXiv preprint arXiv:2212.04983*, 2022. 2

[52] Yihan Wu, Hongyang Zhang, and Heng Huang. Retrievalguard: Provably robust 1-nearest neighbor image retrieval. In *International Conference on Machine Learning*, pages 24266–24279. PMLR, 2022. 2

[53] Dinghuai Zhang, Hongyang R. Zhang, Aaron C. Courville, Yoshua Bengio, Pradeep Ravikumar, and Arun Sai Suggala. Building robust ensembles via margin boosting. In *International Conference on Machine Learning*, pages 26669–26692, 2022. 2

[54] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Accelerating adversarial training via maximal principle. In *Advances in Neural Information Processing Systems*, pages 227–238, 2019. 2

[55] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pages 7472–7482, 2019. 2, 17

[56] Jingfeng Zhang, Xilie Xu, Bo Han, Gang Niu, Li zhen Cui, Masashi Sugiyama, and Mohan S. Kankanhalli. Attacks which do not kill training make adversarial learning stronger. In *International Conference on Machine Learning*, 2020. 2

[57] Yonggang Zhang, Mingming Gong, Tongliang Liu, Gang Niu, Xinmei Tian, Bo Han, Bernhard Schölkopf, and Kun Zhang. Adversarial robustness through the lens of causality. In *International Conference on Learning Representations*, 2022. 2

[58] Haizhong Zheng, Ziqi Zhang, Juncheng Gu, Honglak Lee, and Atul Prakash. Efficient adversarial training with transferable adversarial examples. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1178–1187, 2020. 9

A. Additional Lemmas and Proofs

We analyze the following single-target adversarial training problem

$$\min_{\mathbf{w}} \mathbb{E}_{(x,y)\sim\mathcal{D}} \max(0, 1 - y\mathbf{w}^\top(\mathbf{x} + \boldsymbol{\delta}_p)) \quad \text{s.t.} \|\mathbf{w}\|_2 = 1\,, \tag{4}$$

where $p \in \{1, 2, \infty\}$ is given before the training procedure.

Lemma 6 (lemma D.1 [43]). *The optimal solution $\mathbf{w}^* = (\mathbf{w}_1, ..., \mathbf{w}_{d+1})$ of our optimization problem (Equation (3)) must satisfy $\mathbf{w}_2 = \ldots = \mathbf{w}_{d+1}$ and $\operatorname{sign}(\mathbf{w}_2) = \operatorname{sign}(\mu)$.*

Lemma 7 (lemma D.2 [43]). *The optimal solution $\mathbf{w}^* = (\mathbf{w}_1, \ldots, \mathbf{w}_{d+1})$ of our optimization problem (Equation (3)) must satisfy $\mathbf{w}_1 \le 1/\sqrt{2}$ and $\mathbf{w}_2 = \ldots = \mathbf{w}_{d+1} \ge 1/\sqrt{2d}$.*

Lemma 8. *In the adversarial training framework, for arbitrary step t, if $\epsilon > \mu$ and*

$$p \le 1 - \max\left(\frac{\mathbb{E}[\max(0, 1 - \mathcal{N}((\epsilon-\mu)\sqrt{d}, 1))]}{\mathbb{E}[\max(0, 1 + 1/\sqrt{2}(1+\epsilon) - \mathcal{N}((\epsilon-\mu)\sqrt{\frac{d}{2}}, 0.5))]}, \frac{\mathbb{E}[\max(0, 1 - \mathcal{N}((\epsilon+\mu)\sqrt{d}, 1))]}{\mathbb{E}[\max(0, 1 + 1/\sqrt{2}(1+\epsilon) - \mathcal{N}((\epsilon+\mu)\sqrt{\frac{d}{2}}, 0.5))]} \right),$$

the optimal solution $\mathbf{w}^{t*} = (\mathbf{w}_1^t, \ldots, \mathbf{w}_{d+1}^t)$ of our optimization problem must satisfy $\mathbf{w}_1^t \le 1/\sqrt{2}$ and $\mathbf{w}_2^t = \ldots = \mathbf{w}_{d+1}^t$ and $|\mathbf{w}_2^t| \ge 1/\sqrt{2d}$. Moreover, $\operatorname{sign}(\mathbf{w}_i^t) = -\operatorname{sign}(\mathbf{w}_i^{t+1}), \forall i \in [2, d+1]$.

Lemma 9. *If $z \sim \mathcal{N}(\mu, \sigma^2)$,*

$$\mathbb{E}_z[z\mathbb{I}_{z\ge 0}] = \int_0^\infty z \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{(z-\mu)^2}{2\sigma^2}) dz$$
$$= \frac{\sigma}{\sqrt{2\pi}} \exp(-\frac{\mu^2}{2\sigma^2}) + \frac{\mu}{2}(\operatorname{erf}(\frac{\mu}{\sqrt{2}\sigma}) + 1)\,.$$

Lemma 10. *When $\mu \ge 4/\sqrt{d}$, $\epsilon \ge 2\mu$, and $p \le 0.977$, the optimal solution $\mathbf{w}^{t*} = (\mathbf{w}_1^t, \ldots, \mathbf{w}_{d+1}^t)$ of our optimization problem must satisfy $\mathbf{w}_1^t \le 1/\sqrt{2}$ and $\mathbf{w}_2^t = \ldots = \mathbf{w}_{d+1}^t$ and $|\mathbf{w}_2^t| \ge 1/\sqrt{2d}$.*

Lemma 11. *When $\mathbf{w}_t^1 \le 1/\sqrt{2}$ and $\mathbf{w}_t^2 = \ldots = \mathbf{w}_t^{d+1}$ and $|\mathbf{w}_t^2| \ge 1/\sqrt{2d}$, if $\epsilon_\infty \ge \frac{2}{d}\epsilon_1$ and $\epsilon_\infty \ge \sqrt{\frac{2}{d}}\epsilon_2$, ∞-player dominates 1-player and 2-player. In another word, the training procedure cannot converge.*

Lemma 12. *MAX and MSD are the same under the SVM scenario.*

Standard classification is easy. Remind that the data consists of a robust feature $\mathbf{x}_1$, which is strongly related to the label and d non-robust features $\mathbf{x}_i, i \in [2, d+1]$, which are weakly related to the label y. But with the non-robust features, we can construct a simple linear classifier f that achieves over 99% natural accuracy as

$$f(\mathbf{x}) = \operatorname{sign}([0, \frac{1}{d}, \ldots, \frac{1}{d}]^\top \mathbf{x})\,.$$

For the natural accuracy, we have

$$Pr[f(\mathbf{x}) = y] = Pr[\operatorname{sign}([0, \frac{1}{d}, \ldots, \frac{1}{d}]^\top \mathbf{x}) = y]$$
$$= Pr[\frac{y}{d}\sum_{i=1}^{d} \mathcal{N}(\mu y, 1) > 0] = Pr[\mathcal{N}(\mu, \frac{1}{d}) > 0] \ge 0.99\,,$$

when $\mu \ge \frac{3}{\sqrt{d}}$.

Robust classification is not easy. We have the opposite observation when facing ℓ_∞ adversarial training. The robust accuracy is shown as

$$\min_{\|\boldsymbol{\delta}_\infty\|_\infty \le \epsilon_\infty} Pr[f(\mathbf{x} + \boldsymbol{\delta}_\infty) = y] \le Pr[\mathcal{N}(\mu, \frac{1}{d}) - \epsilon > 0]$$
$$= Pr[\mathcal{N}(-\mu, \frac{1}{d}) > 0] \le 0.01\,,$$

when $\epsilon_\infty = 2\mu$. In the following part of our paper, we show that it is not only difficult to get a fairly good robust accuracy, but also a converged model under the multi-target adversarial training problem.

B. Proofs

B.1. Proof of Theorem 1

Proof. Combining Lemma 8, Lemma 11, and Lemma 12 yields this theorem. □

B.1.1 Proof of Theorem 3

Proof. As the ∞-player dominates this game, the multi-target adversarial training problem reduces to the single-target problem equation 4. Further, with Lemma 8, for non-robust feature i, at any time t, we have $\operatorname{sign}(\mathbf{w}_i^t) = -\operatorname{sign}(\mathbf{w}_i^{t-1})$. Thus the training procedure does not converge. □

B.1.2 Proof of Theorem 4

Proof. For the i-th player's loss (the i-th player dominates the bargaining game at the time t), as the loss function is μ-strongly convex, we have

$$\ell_i(\mathbf{w}^{t+1}) \ge \ell_i(\mathbf{w}^t) - \ell_i'(\mathbf{w}^t)^\top(\mathbf{w}^{t+1} - \mathbf{w}^t) + \frac{\mu}{2}\|\mathbf{w}^{t+1} - \mathbf{w}^t\|_2^2$$
$$\ell_i(\mathbf{w}^{t+1}) \ge \ell_i(\mathbf{w}^t) + \eta\ell_i'(\mathbf{w}^t)^\top\ell_i'(\mathbf{w}^t) + \frac{\mu\eta^2}{2}\|\ell_i'(\mathbf{w}^t)\|_2^2 > \ell_i(\mathbf{w}^t)\,.$$

For the j-th player's loss and $j \neq i$, as the loss function is μ-strongly convex, we have

$$\ell_j(\mathbf{w}^{t+1}) \geq \ell_j(\mathbf{w}^t) - \ell_j^{'}(\mathbf{w}^t)^\top(\mathbf{w}^{t+1} - \mathbf{w}^t) + \frac{\mu}{2}\|\mathbf{w}^{t+1} - \mathbf{w}^t\|_2^2$$

$$\ell_j(\mathbf{w}^{t+1}) \geq \ell_j(\mathbf{w}^t) + \frac{\mu\eta^2}{2}\|\ell_i^{'}(\mathbf{w}^t)\|_2^2 > \ell_j(\mathbf{w}^t)\,.$$

That means at time t, the loss of all player will keep increasing. And thus, if one player dominate the bargaining game throughout the whole game, the loss of all players and will keep increasing during the whole game, which means the bargaining game might not converge. □

B.1.3 Proof of Theorem 5

Proof. As the loss function is L-smooth, $\forall i$, we have

$$\begin{aligned}
\ell_i(\mathbf{w}^{t+1}) \leq& \ell_i(\mathbf{w}^t) + \eta\ell_i(\mathbf{w}^t)^\top(\mathbf{w}^{t+1} - \mathbf{w}^t) \\
&+ \frac{L}{2}\|\eta\sum_{k\in[K]} g_k^t/K\|^2, \quad \text{(as L-smooth)} \\
\ell_i^{t+1} \leq& \ell_i^t - \eta g_i^{t\top}\sum_{k\in[K]} g_k^t/K + \frac{L}{2}\|\eta\sum_{k\in[K]} g_k^t/K\|^2, \\
=& \ell_i^t - \eta g_i^{t\top} g_i/K + \frac{L\eta^2}{2K^2}\sum_{k\in[K]} g_k^{t\top} g_k\,.
\end{aligned}$$

Summing the above inequality from $i = 1$ to K, we have

$$\ell^{t+1} \leq \ell^t - \frac{\eta}{K}\sum_{k\in[K]} g_k^{t\top} g_k + \frac{L\eta^2}{2K}\sum_{k\in[K]} g_k^{t\top} g_k < \ell^t\,. \quad (\text{as } \eta < \frac{2}{L}) \tag{5}$$

The proof is completed. □

B.2. Proof of Theorem 8

$t = 0$, by Theorem 7 the result holds and $\text{sign}(\mathbf{w}_i^0) = \text{sign}(\mu), \forall i \in [2, d+1]$.

$t = 1$, the perturbed distribution is given by

$$y \sim \{-1, 1\}, \quad \mathbf{x}_1 \sim \begin{cases} y(1-\epsilon), & \text{with prob } p; \\ -y(1+\epsilon), & \text{with prob } 1-p, \end{cases}$$
$$\mathbf{x}_i \sim \mathcal{N}((\mu-\epsilon)y, 1),\ i \geq 2$$

Assume for the sake of contradiction that $\mathbf{w}_1^1 \geq 1/\sqrt{2}$, by Theorem 6 we have $0 \geq \mathbf{w}_2^1 = \ldots = \mathbf{w}_{d+1}^1 \geq -1/\sqrt{2d}$. Then, with probability at least $1-p$, the first feature predicts the wrong label and without enough weight, the remaining features cannot compensate for it. Concretely,

$$\begin{aligned}
&\mathbb{E}[\max(0, 1 - y\mathbf{w}^{1*\top}(\mathbf{x} - \boldsymbol{\delta}_\infty))] \\
\geq& (1-p)\mathbb{E}[\max(0, 1 + \mathbf{w}_1^1(1+\epsilon) - |\mathbf{w}_2^1|\sum_{i=2}^{d+1}\mathcal{N}(\epsilon-\mu, 1))] \\
\geq& (1-p)\mathbb{E}[\max(0, 1 + 1/\sqrt{2}(1+\epsilon) - \mathcal{N}((\epsilon-\mu)\sqrt{\frac{d}{2}}, 0.5))]
\end{aligned}$$

We will now show that a solution that assigns zero weight on the first feature ($\mathbf{w}_2^1 = 1/\sqrt{d}$ and $\mathbf{w}_1^1 = 0$), achieves a better margin loss,

$$\begin{aligned}
&\mathbb{E}[\max(0, 1 - y\mathbf{w}_1^\top(\mathbf{x} - \boldsymbol{\delta}_\infty))] \\
=& \mathbb{E}[\max(0, 1 - \mathcal{N}((\epsilon-\mu)\sqrt{d}, 1))]\,.
\end{aligned}$$

Because

$$p \leq 1 - \frac{\mathbb{E}[\max(0, 1 - \mathcal{N}((\epsilon-\mu)\sqrt{d}, 1))]}{\mathbb{E}[\max(0, 1 + 1/\sqrt{2}(1+\epsilon) - \mathcal{N}((\epsilon-\mu)\sqrt{\frac{d}{2}}, 0.5))]},$$

we have $\mathbb{E}[\max(0, 1 - y\mathbf{w}^{1*\top}(\mathbf{x} - \boldsymbol{\delta}_\infty))] \geq \mathbb{E}[\max(0, 1 - y\mathbf{w}^{1\top}(\mathbf{x} - \boldsymbol{\delta}_\infty))]$, which yields contradiction. Besides, in this case $\text{sign}(\mathbf{w}_i^1) = \text{sign}(\mu - \epsilon) = -\text{sign}(\mu) = -\text{sign}(\mathbf{w}_i^0), \forall i \in [2, d+1]$

$t = 2$, the perturbed distribution is given by

$$y \sim \{-1, 1\}, \quad \mathbf{x}_1 \sim \begin{cases} y(1-\epsilon), & \text{with prob } p; \\ -y(1+\epsilon), & \text{with prob } 1-p, \end{cases}$$
$$\mathbf{x}_i \sim \mathcal{N}((\mu+\epsilon)y, 1),\ i \geq 2\,.$$

Assume for the sake of contradiction that $\mathbf{w}_1^2 \geq 1/\sqrt{2}$, by Theorem 6 we have $0 \geq \mathbf{w}_2^2 = \ldots = \mathbf{w}_{d+1}^2 \geq -1/\sqrt{2d}$. Then, with probability at least $1-p$, the first feature predicts the wrong label and without enough weight, the remaining features cannot compensate for it. Concretely,

$$\begin{aligned}
&\mathbb{E}[\max(0, 1 - y\mathbf{w}^{2*\top}(\mathbf{x} - \boldsymbol{\delta}_\infty))] \\
\geq& (1-p)\mathbb{E}[\max(0, 1 + \mathbf{w}_1^2(1+\epsilon) - |\mathbf{w}_2^2|\sum_{i=2}^{d+1}\mathcal{N}(\epsilon+\mu, 1))] \\
\geq& (1-p)\mathbb{E}[\max(0, 1 + 1/\sqrt{2}(1+\epsilon) - \mathcal{N}((\epsilon+\mu)\sqrt{\frac{d}{2}}, 0.5))]\,.
\end{aligned}$$

We will now show that a solution that assigns zero weight on the first feature ($\mathbf{w}_2^2 = 1/\sqrt{d}$ and $\mathbf{w}_1^2 = 0$), achieves a better margin loss.

$$\begin{aligned}
&\mathbb{E}[\max(0, 1 - y\mathbf{w}_2^\top(\mathbf{x} - \boldsymbol{\delta}_\infty))] \\
=& \mathbb{E}[\max(0, 1 - \mathcal{N}((\epsilon+\mu)\sqrt{d}, 1))]\,.
\end{aligned}$$

Because

$$p \leq 1 - \frac{\mathbb{E}[\max(0, 1 - \mathcal{N}((\epsilon+\mu)\sqrt{d}, 1))]}{\mathbb{E}[\max(0, 1 + 1/\sqrt{2}(1+\epsilon) - \mathcal{N}((\epsilon+\mu)\sqrt{\frac{d}{2}}, 0.5))]},$$

we have $\mathbb{E}[\max(0, 1 - y\mathbf{w}^{2*\top}(\mathbf{x} - \boldsymbol{\delta}_\infty))] \geq \mathbb{E}[\max(0, 1 - y\mathbf{w}^{2\top}(\mathbf{x} - \boldsymbol{\delta}_\infty))]$, which yields contradiction. Besides, in this case $\text{sign}(\mathbf{w}_i^2) = \text{sign}(\mu + \epsilon) = \text{sign}(\mu) = -sign(\mathbf{w}_i^1), \forall i \in [2, d+1]$

By induction we can easily derive that $\mathbf{w}_1^t \leq 1/\sqrt{2}$, $\mathbf{w}_2^t = \ldots = \mathbf{w}_{d+1}^t$, $|\mathbf{w}_2^t| \geq 1/\sqrt{2d}$ and $\text{sign}(\mathbf{w}_i^t) = -\text{sign}(\mathbf{w}_i^{t+1}), \forall i \in [2, d+1], \forall t \geq 0$.

B.3. Proof of Lemma 10

Let $\mu = m/\sqrt{d}, m \geq 4, \epsilon = k\mu, k \geq 2$.

We have,

$$\begin{aligned}
&\frac{\mathbb{E}[\max(0, 1 - \mathcal{N}((\epsilon - \mu)\sqrt{d}, 1))]}{\mathbb{E}[\max(0, 1 + 1/\sqrt{2}(1+\epsilon) - \mathcal{N}((\epsilon - \mu)\sqrt{\frac{d}{2}}, 0.5))]} \\
=&\frac{\mathbb{E}[\max(0, \mathcal{N}(1 + m - km, 1))]}{\mathbb{E}[\max(0, \mathcal{N}(1 + (1+m)/\sqrt{2} + km/\sqrt{2d} - km/\sqrt{2}, 0.5))]} \\
\leq&\frac{\mathbb{E}[\max(0, \mathcal{N}(1 + m - km, 1))]}{\mathbb{E}[\max(0, \mathcal{N}(1 + (1 + m - km)/\sqrt{2}, 0.5))]}\,.
\end{aligned}$$

Consider the function $h(a) = \frac{\mathbb{E}[\max(0, \mathcal{N}(a, 1))]}{\mathbb{E}[\max(0, \mathcal{N}(1+a/\sqrt{2}, 0.5))]} = \frac{\frac{1}{\sqrt{2\pi}}\exp(-\frac{a^2}{2}) + \frac{a}{2}(\text{erf}(\frac{a}{\sqrt{2}}) + 1)}{\frac{1}{2\sqrt{\pi}}\exp(-(1+\frac{a}{\sqrt{2}})^2) + \frac{1+\frac{a}{\sqrt{2}}}{2}(\text{erf}((1+\frac{a}{\sqrt{2}})) + 1)}$.

We have,

$$\begin{aligned}
h'(a) =&((\frac{1}{2} + \frac{1}{2}\text{erf}(\frac{a}{\sqrt{2}}))(\frac{1}{2\sqrt{\pi}}\exp(-(1 + \frac{a}{\sqrt{2}})^2) \\
&+ \frac{1 + \frac{a}{\sqrt{2}}}{2}(\text{erf}((1 + \frac{a}{\sqrt{2}})) + 1)) \\
&- (\frac{1}{2\sqrt{2}} + \frac{1}{2\sqrt{2}}\text{erf}(1 + \frac{a}{\sqrt{2}}))(\frac{1}{\sqrt{2\pi}}\exp(-\frac{a^2}{2}) \\
&+ \frac{a}{2}(\text{erf}(\frac{a}{\sqrt{2}}) + 1))))/(\frac{1}{2\sqrt{\pi}}\exp(-a^2) \\
&+ \frac{a}{2}(\text{erf}(a) + 1))^2\,.
\end{aligned}$$

By numerical simulation we have $h'(a) \geq 0$, when $a \leq 0$, so $h(a)$ is increasing with a when $a \leq 0$, thus

$$\begin{aligned}
&1 - \max(\frac{\mathbb{E}[\max(0, 1 - \mathcal{N}((\epsilon - \mu)\sqrt{d}, 1))]}{\mathbb{E}[\max(0, 1 + 1/\sqrt{2}(1+\epsilon) - \mathcal{N}((\epsilon - \mu)\sqrt{\frac{d}{2}}, 0.5))]} \\
&\frac{\mathbb{E}[\max(0, 1 - \mathcal{N}((\epsilon + \mu)\sqrt{d}, 1))]}{\mathbb{E}[\max(0, 1 + 1/\sqrt{2}(1+\epsilon) - \mathcal{N}((\epsilon + \mu)\sqrt{\frac{d}{2}}, 0.5))]}) \\
\geq&1 - h(-3) = 0.9775 > p.
\end{aligned}$$

By Lemma 8, we have the optimal solution $\mathbf{w}^{t*} = (\mathbf{w}_1^t, \ldots, \mathbf{w}_{d+1}^t)$ of our optimization problem must satisfy $\mathbf{w}_1^t \leq 1/\sqrt{2}$ and $\mathbf{w}_2^t = \ldots = \mathbf{w}_{d+1}^t$ and $|\mathbf{w}_2^t| \geq 1/\sqrt{2d}$.

B.4. Proof of Lemma 11

Let $\ell_p = 1 - yw^\top(\mathbf{x} + \boldsymbol{\delta}_p)$, we have

$$\begin{aligned}
\ell_\infty - \ell_1 =&y\mathbf{w}_t^\top(\boldsymbol{\delta}_1 - \boldsymbol{\delta}_\infty) = \epsilon_\infty\|\mathbf{w}\|_1 - \epsilon_1\frac{\|\mathbf{w}_t\|_2^2}{\|\mathbf{w}_t\|_1} \\
&\geq \epsilon_1(\frac{2}{d}||\mathbf{w}_t||_1^2 - 1) \\
&\geq \epsilon_1(\frac{2}{d}(|\mathbf{w}_t^1| + d|\mathbf{w}_t^2|)^2 - 1) \\
&\geq \epsilon_1(\frac{2}{d}(\frac{1}{\sqrt{2}} + d\frac{1}{\sqrt{2d}})^2 - 1) > 0\,., \\
\ell_\infty - \ell_2 =&y\mathbf{w}^\top(\boldsymbol{\delta}_2 - \boldsymbol{\delta}_\infty) = \epsilon_\infty\|\mathbf{w}\|_1 - \epsilon_1\frac{\|\mathbf{w}\|_2^2}{\|\mathbf{w}\|_2} \\
&\geq \epsilon_2(\sqrt{\frac{2}{d}}||\mathbf{w}_t||_1 - 1) \\
&\geq \epsilon_2(\sqrt{\frac{2}{d}}(|\mathbf{w}_t^1| + d|\mathbf{w}_t^2|) - 1) \\
&\geq \epsilon_2(\sqrt{\frac{2}{d}}(\frac{1}{\sqrt{2}} + d\frac{1}{\sqrt{2d}}) - 1) > 0\,.
\end{aligned}$$

Now, we have proved that ∞-player dominates others and $\text{sign}(\mathbf{w}_i^t) = -\text{sign}(\mathbf{w}_i^{t-1})$. With Lemma 8, we know that at any time t, we have $|\mathbf{w}_t^i - \mathbf{w}_{t-1}^i| \geq \sqrt{1/d}, \forall i \in [2, d+1]$, which means the training procedure cannot converge.

B.5. Proof of Lemma 12

Under the deep learning cases (non-linear and non-convex), MSD follows the steepest direction (ℓ_1, ℓ_2 or ℓ_∞) in each PGD step to find the perturbation which approximately maximizes the loss function, while MAX uses PGD to find the perturbations empirically and then chooses the perturbation maximizing the loss function. MSD and MAX are different approaches in deep learning cases (non-linear and non-convex).

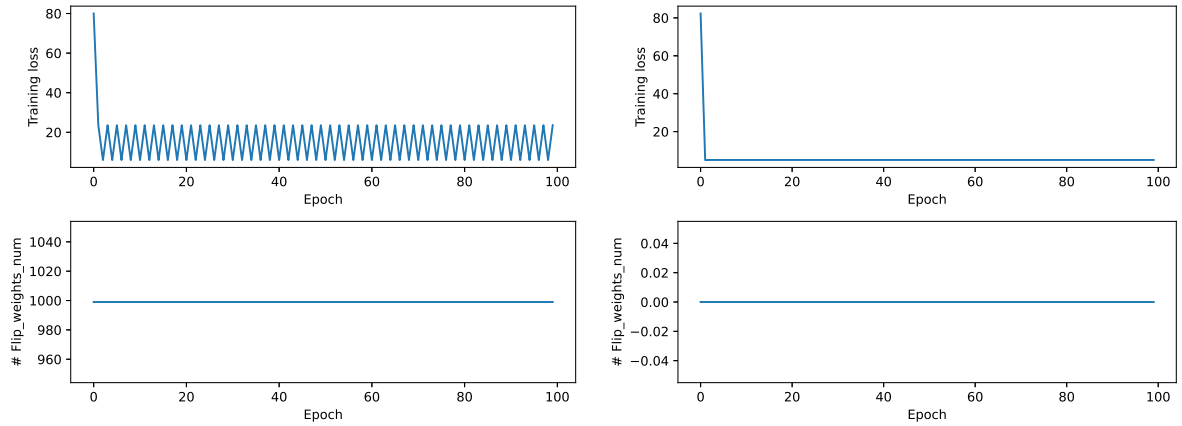
On the other side, under the SVM (convex and linear) case, the optimal perturbations with ℓ_1, ℓ_2 and ℓ_∞ constraints have analytical solutions. In this way, both MSD and MAX can directly determine which perturbation maximizes the loss within one step, which means MSD and MAX are the same under the SVM case.

C. Extra Experiments

Due to the limitation of space, we put the experimental verification of our negative results and Figure 6 here.

C.1. Verifying effects of player domination on the SVM case

To verify our theoretical results, we conduct experiments and the corresponding results are shown in Figure 3. We use a fully connected network (Fully Connected Layer (in=d, out=1), where $d = 1000$). For the data generation, we set



(a) MAX and MSD (b) AVG

Figure 3. We illustrate the training loss and the number of weights that flip between two epochs. Figure 3a shows the data of model trained with MAX and MSD using the SVM model (Sec 4.1) while Figure 3b shows the number of model trained with AVG.

$p = 0.95$, $\mu = 4/\sqrt{d}$, and $\epsilon_1 = \epsilon_2 = \epsilon_\infty = 2\mu$, and the sample size is 100000.

We notice that with MAX or MSD (they are equal under the SVM scenario, Lemma 12), the training procedure cannot converge as the training loss is fluctuating while the number of weights whose signs are flipped compared with last epoch is almost 1000. At the same time, AVG does converge. That complements our theoretical results (Theorem 1).

Besides, we also conduct experiments verifying the conjecture that when ℓ_1 or ℓ_2 player dominates the bargaining game, the training procedure does not converge as well. For the case when ℓ_1 dominates, we set $\epsilon_1 = 4\mu, \epsilon_2 = \epsilon_\infty = 2\mu$, while when ℓ_2 dominates, we set $\epsilon_2 = 4\mu, \epsilon_1 = \epsilon_\infty = 2\mu$. We observe exactly the same curves as Figure 3, showing that when ℓ_1 and ℓ_2 dominates, the training procedure with MAX and MSD cannot converge while with AVG, training procedure does converge. We present the results in Figures 4 and 5.

C.2. Results on CIFAR10

Similar, for the CIFAR10 dataset, we require that the adapted epsilon are bigger than the half of and smaller that twice of the original epsilon.

Robustness curves. The robustness curves are shown in Figures 6. The lines of MAX with either ℓ_1 or ℓ_2 norm-based AdaptiveBudget are higher than the lines without AdaptiveBudget. The gap between lines with the adaptive budget method and lines without is biggest when the budget of the adversary is small.

Norm choice in adaptive budget. We notice that the choice of norm in the adaptive budget barely influences the robust accuracy as shown in Table 2. On both three methods, *i.e.*, MAX, MSD, and AVG, our proposed adaptive budget is able to improve the performance with both ℓ_1 and ℓ_2 norms, while the difference between ℓ_1 and ℓ_2 norm is only 0.6% and 1.7%, 0.4% and 2.8%, 1.3% and 0.9% on MAX, MSD, and AVG against PGD and AutoAttack adversaries.

Table 4. The robust accuracy of TRADES on MNIST and CIFAR-10.

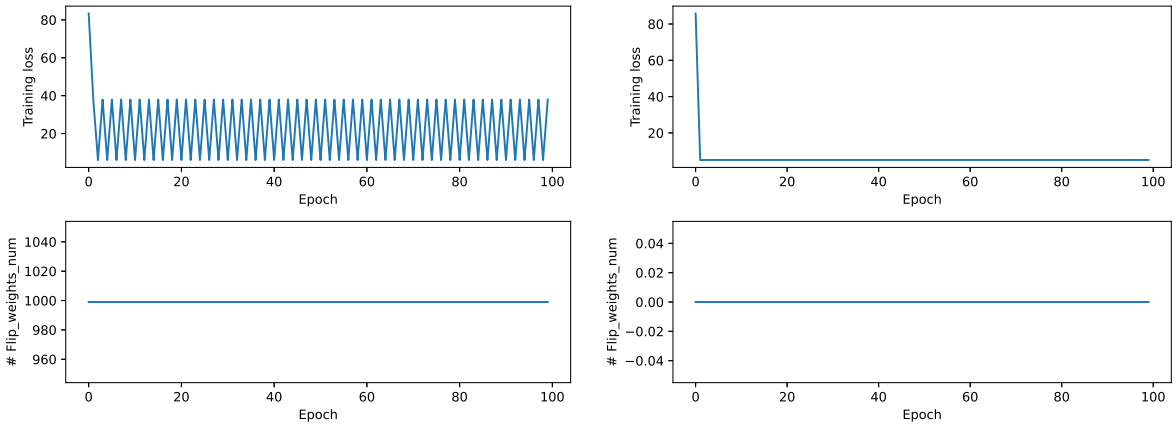
TRADES	MNIST	CIFAR-10
ℓ_∞ AA	90.9	55.1
ℓ_1 AA	4.6	6.2
ℓ_2 AA	11.2	60.5
All AA	3.7	6.2

C.3. Verification of avoiding player domination

We record the maximum number of consecutive steps during which the gradient of a player is consistently higher than others on MNIST with deep neural networks. Quantitative results in Table 5 indicate that, with *AdaptiveBudget* on MNIST, no player is able to control the updates for more than 1207 steps, whereas, with MSD, the ℓ_∞-player controlled the updates for 6891 consecutive steps, respectively. Besides, the average number during which a player dominated is significantly decreased with *AdaptiveBudget*. This demonstrates the effectiveness of *AdaptiveBudget* in avoiding player domination.

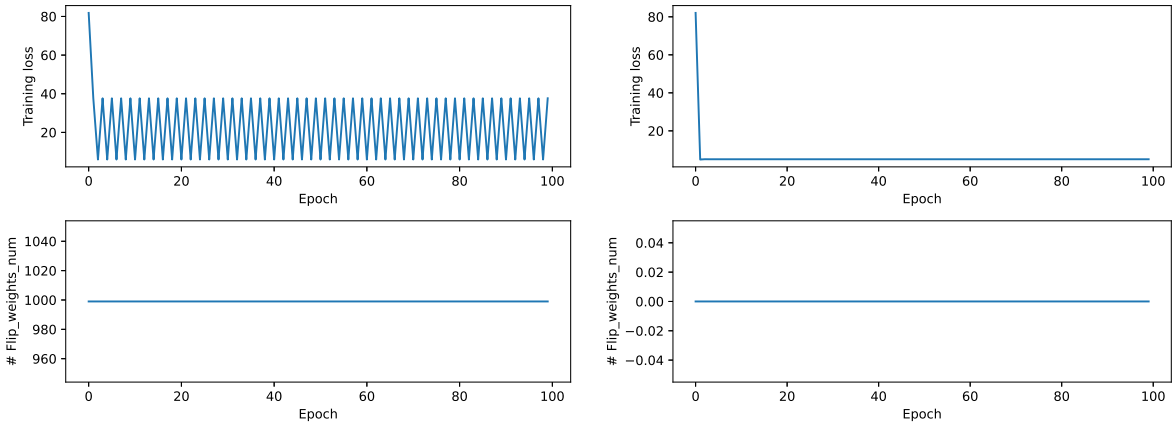
C.4. Are single-target robustness algorithms able to achieve multi-target robustness?

In Tables 1 and 2, we illustrate that three typical single-target methods, *i.e.*, ℓ_1, ℓ_2, and ℓ_∞, are not able to maintain multi-target robustness as the overall robust accuracy drops a



(a) MAX and MSD (b) AVG

Figure 4. We illustrate the training loss and the number of weights that flip between two epochs. We set $\epsilon_1 = 4\mu, \epsilon_2 = \epsilon_\infty = 2\mu$. Figure 4a shows the data of model trained with MAX and MSD using the SVM model (Sec 4.1) while Figure 4b shows the number of model trained with AVG.



(a) MAX and MSD (b) AVG

Figure 5. We illustrate the training loss and the number of weights that flip between two epochs. We set $\epsilon_2 = 4\mu, \epsilon_1 = \epsilon_\infty = 2\mu$. Figure 5a shows the data of model trained with MAX and MSD using the SVM model (Sec 4.1) while Figure 5b shows the number of model trained with AVG.

Table 5. This table displays the maximum number of consecutive steps (updates) during which the gradient of a single player is bigger than others on MNIST. "w. ℓ_1" and "w. ℓ_2" refer to the algorithm with *AdaptiveBudget*. "AVG" refers to the average length of consecutive steps that player domination lasts. For example, if the training process consists of 5 steps and the dominant players are $[\ell_1, \ell_1, \ell_2, \ell_\infty, \ell_\infty]$, the longest consecutive steps for ℓ_1, ℓ_2, and ℓ_∞ are 2, 1, and 2, and the average step (AVG) is 1.67.

	MSD			MAX			AVG		
		w. ℓ_1	w. ℓ_2		w. ℓ_1	w. ℓ_2		w. ℓ_1	w. ℓ_2
ℓ_1-PGD adversary	60	61 ↑	27 ↓	15	5 ↓	9 ↓	8	10 ↑	4 ↓
ℓ_2-PGD adversary	113	140 ↑	209 ↑	337	323 ↓	303 ↓	291	154 ↓	39 ↓
ℓ_∞-PGD adversary	6891	1207 ↓	572 ↓	52	45 ↓	26 ↓	32	26 ↓	123 ↑
AVG	18.77	10.44 ↓	8.97 ↓	6.23	2.41 ↓	2.41 ↓	4.24	2.45 ↓	2.33 ↓

lot. To compare with more methods, we conduct experiments on a representative single-target method, *i.e.*, TRADES [55]. We found that TRADES fails to defend ℓ_1, ℓ_2, and ℓ_∞ AutoAttack's adversaries simultaneously. As shown in Table 4, the overall accuracies of TRADES on MNIST and CIFAR-10 are only 3.9% and 6.2%, respectively.

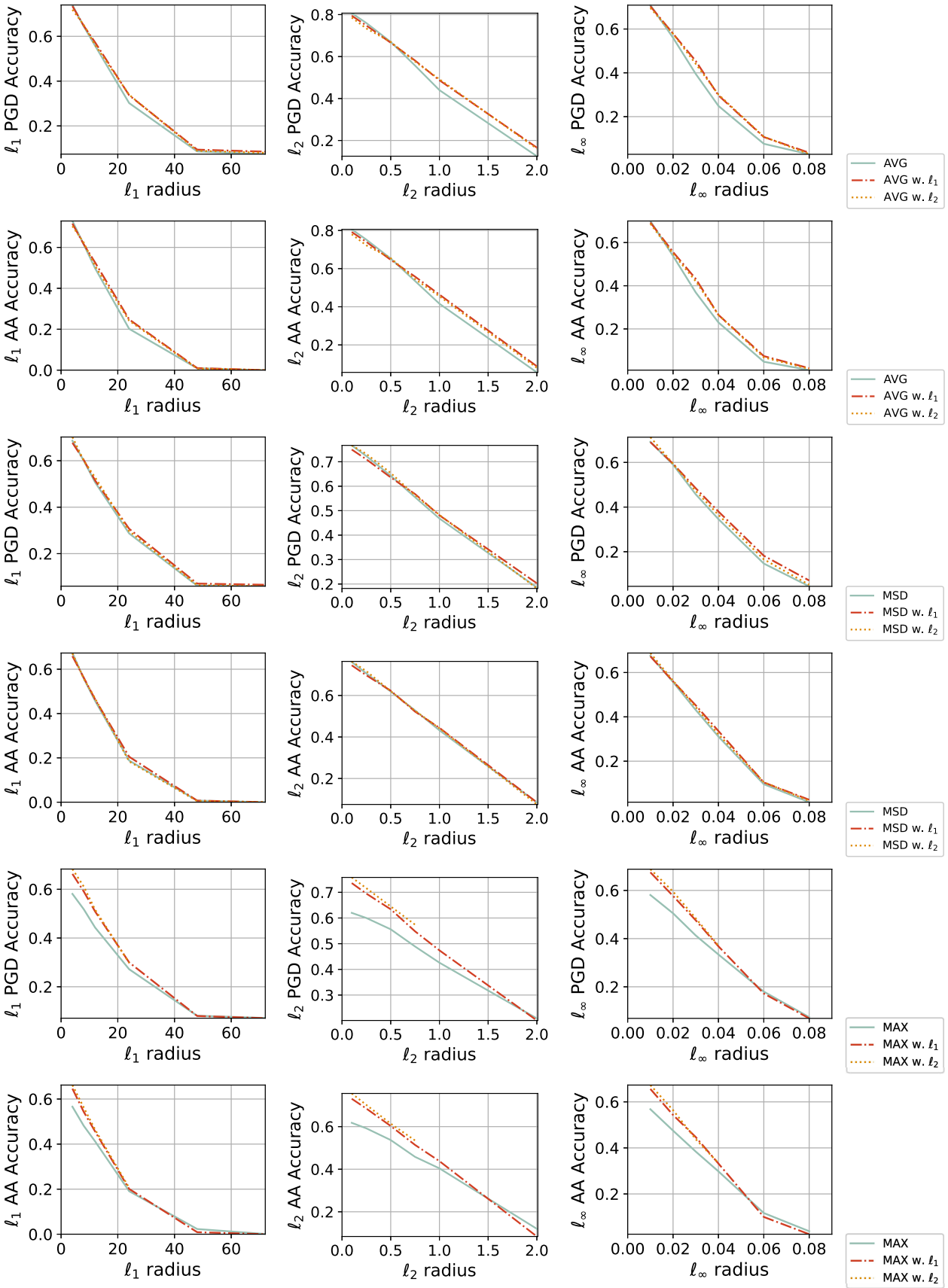


Figure 6. Robustness curves show the adversarial accuracy on CIFAR-10 trained with MSD, AVG, and MAX against ℓ_1 (left), ℓ_2 (middle), and ℓ_∞ (right) PGD and AutoAttack ("AA" in the figures) adversaries over a range of epsilon. "w. ℓ_1" and "w. ℓ_2" are methods with AdaptiveBudget using ℓ_1 or ℓ_2 norms.