TEST-TIME ADAPTATION AND ADVERSARIAL ROBUSTNESS

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper studies test-time adaptation in the context of adversarial robustness. We first formulate an adversarial threat model for test-time adaptation, where the defender may have a unique advantage as the adversarial game becomes a maximin game, instead of a minimax game as in the classic adversarial robustness threat model. On the positive side, we show that, if one is allowed to access the training data, then *domain adversarial training*, such as DANN, can provide nontrivial robustness in the test-time threat model, even against strong transfer attacks and adaptive fixed point attacks. This is somewhat surprising as DANN is not designed specifically for adversarial robustness (e.g. against norm-based attacks). On the negative side, we show that recent data-oblivious test-time adaptations have poor adversarial robustness. We take a step to discuss moving towards adversarially robust test-time adaptation and examine its various implications.

1 INTRODUCTION

There is a surge of interest to study test-time adaptation to help generalization to unseen domains (e.g., recent work by Sun et al. (2020); Wang et al. (2020); Nado et al. (2020)). At the high level, a generic test-time adaptation can be modeled as an algorithm Γ which accepts an (optional) labeled training dataset D, an (optional) model F trained on D (usually used as a starting point), and an unlabeled test feature set U, outputs a model $\widetilde{F} = \Gamma(F, D, U)$, in order to achieve high test accuracy on U. For large test set U, test-time adaptation can be viewed as a form of *transductive learning* proposed by Vapnik (1998) (i.e., using D, U to train a model to predict specific instances in U), which is argued to be easier than more traditional inductive learning.

This paper studies test-time adaptation in the context of adversarial robustness (i.e., there is an active agent who tries to fool the test-time adaptation so that $\widetilde{F}$ gives wrong predictions). There are several motivations in pursuing this direction. First, this question (in particular for large U) is of practical interest: Many practical ML pipelines run in a batch mode[1], where they first collect a set of unlabelled data points, and then send them to a model (e.g. Nado et al. (2020)). In such cases, data in the batch may have been adversarially perturbed, and it is a natural question *whether we can leverage the large batch size and test-time adaptation to enhance adversarial robustness.* Second, from a purely theoretical point of view, since test-time adaptation is a form of transductive learning, it is intriguing to ask *whether transductive adversarial learning can be easier*, given that traditional adversarial robustness is formulated in the *inductive* learning setting (e.g. Madry et al. (2018)). To this end, a recent theory work by Goldwasser et al. (2020) shows that, with transductive learning, one can achieve nontrivial guarantees for classes of bounded VC dimension with *arbitrary* train and test distributions. The current work complements their paper in the setting of deep learning.

To study these questions, we first formalize a threat model, which we call *(test-time) maximin threat model*, for the adversarial robustness of test-time adaptation. Roughly speaking, while the classic adversarial robustness game is a minimax game $\min_F \mathbb{E}_V[\max_{\widetilde{V}} L(F, \widetilde{V})]$, the maximin threat model gives a maximin game: $\mathbb{E}_V[\max_{\widetilde{V}} \min_{\widetilde{F}} L(\widetilde{F}, \widetilde{V})]$. By the maximin inequality, it follows immediately that this threat model is *no harder* than the minimax model (strictly speaking, since our maximin threat model allows source and target domain to be different, we need to generalize the classic minimax model to such a case, see Definition 4).

[1]For example, Instgram collects a large batch of photos before sending them to a model to tag persons.

We move on to study whether the maximin threat model is "strictly easier" than the minimax threat model. We note that any good defender solution (a robust model) in the minimax game induces a good defender solution in the maximin game (an adaptation algorithm that outputs that robust model), thus intuitively, the good defender solutions of the minimax model is a subset of the good defender solutions of the maximin threat model. We ask *whether such a containment is proper*: That is, whether there is a defender solution that is good in the maximin threat model (i.e., provides robustness), but is bad in the minimax threat model (provides no robustness). A positive answer will demonstrate that the maximin threat model admits more good solutions. Besides theoretical interest, this question is also of practical importance since these "new" solutions may possess *desirable properties that good solutions in the minimax threat model may lack*. For example, such solutions may be computationally more feasible than minimax solutions, as various previous work have argued the hardness of minimax optimization Katz et al. (2017); Daskalakis et al. (2020).

On the positive side, we consider ℓ_∞ attacks, and empirically show that DANN (Ganin et al. (2017)), an algorithm designed for *unsupervised domain adaptation* (UDA), is such a candidate. This is somewhat surprising as DANN is not specifically designed for ℓ_∞ attacks (as desribed by Goodfellow (2018) (pp.30), a desirable defender solution is not to directly optimize the performance measure.) Specifically, we generate an adapted model $\widetilde{F} = \mathsf{DANN}(\mathsf{RI}, D, U)$ to classify U, where RI is random initialization, D is the labeled training set, and U is generated by either transfer attacks using PGD attacks, or adaptive fixed point attacks. On CIFAR10 for example, this achieves about 50% test accuracy, comparable with robustness achieved by an adversarially trained model (specifically for ℓ_∞ attacks) in the minimax threat model. Not suriprisingly, the accuracy becomes very low when we evaluate $\widetilde{F}$ in the minimax threat model.

On the negative side, we explore the maximin robustness of the recent data-oblivious adaptation algorithms (namely, the adaptation algorithm does not use D, but just the pretrained model F and unlabeled test set U). Specifically, we consider Test-Time Training (TTT) by Sun et al. (2020). We show that TTT can be easily attacked using simple transfer attacks.

The rest of the paper is organized as follows: Section 2 presents necessary setup. In Section 3 and 4 we present the maximin threat model, and adversarial semi-supervised minimax threat model (a generalization of the classic minimax threat model). In Section 5 we show that DANN provides, somewhat surprisingly, adversarial robustness in the maximin model. Finally, Section 6 explores the maximin robustness of oblivious test-time adaptation, and concludes the paper with future directions.

2 PRELIMINARIES

Threat model for classic adversarial robustness. To formulate the threat model for test-time adaptation, we first present a threat model for the classic adversarial robustness. Although the classic adversarial robustness can be written down succinctly as a minimax objective, namely $\min_F \mathbb{E}_{(x,y)\sim(X,Y)} \left[\max_{x'\in N(x)}[\ell(F;x',y)]\right]$ ($N(x)$ is a neighborhood function of x, determined by the attack type), a threat model formulation will help us develop more nuanced models.

Definition 1 (**Threat model for classic adversarial robustness).** *Attacker and defender agree on a particular attack type (e.g., ℓ_∞ attacks with budget ε). Attacker is an algorithm $\mathcal{A}$, and defender is a supervised learning algorithm $\mathcal{T}$. The game protocol is as follows:*

Before game starts

- *A (labeled) training set D is sampled i.i.d. from from (X, Y).*

Training time

- ***(Defender)** Train a model F on D as $F = \mathcal{T}(D)$.*

Test time

- *A (labeled) natural test set V is sampled i.i.d. from (X, Y).*
- ***(Attacker)** On input F (defender's model), D (source training set), and V (clean test set), $\mathcal{A}$ perturbs each point $(x, y) \in V$ to (x', y) (subject to the agreed attack type), giving $\widetilde{V} = \mathcal{A}(F, D, V)$.*

Evaluation: *Evaluate the test loss of F on $\widetilde{V}$, $L(F, \widetilde{V}) = \frac{1}{|\widetilde{V}|}\sum_{(x',y)\in\widetilde{V}} \ell(F; x', y)$. Attacker's goal is to maximize the test loss, while the defender's goal is to minimize the test loss. if $\ell(F; x', y)$ is the binary loss $\mathbb{1}\{F(x') \neq y\}$, this is the test error of F on $\widetilde{V}$.*

We stress that the i.i.d sampling of V is important (which is also present in the expectation in the minimax objective): Otherwise an attacker can pick any point that fools F and repeat it arbitrarily many times. (we refer readers to Goodfellow (2019) for more discussions along this line).

3 TEST-TIME MAXIMIN THREAT MODEL

The intuition behind the test-time maximin threat model is as follows: After we receive the adversarially perturbed data U to classify (at test time), the defender trains a model based on U, and we evaluate the test accuracy only for U. (i.e. for different test set U we may have different models and different test accuracies.) This perspective of training a model using labeled data D and unlabeled data U, and only test on U, is essentially the *transductive learning* by Vapnik Vapnik (1998) (however, we considered it in an adversarial setting). Fomrally, we have the following definition:

Definition 2 (**Maximin threat model for test-time adaptation).** *Fix an adversarial perturbation type (e.g. ℓ_∞ attacks with norm budget ε). We have a source domain (X^s, Y^s), and a target domain (X^t, Y^t). Both domains are clean data, and are not related to adversarial perturbations. Attacker is an algorithm $\mathcal{A}$, and defender is a pair of algorithms $(\mathcal{T}, \Gamma)$, where $\mathcal{T}$ is a supervised learning algorithm, and Γ is a test-time adaptation algorithm. The game protocol is as follows:*

Before game starts

- *A (labeled) training set D is sampled i.i.d. from (X^s, Y^s).*

Training time

- **(Defender, optional)** *Train a model F trained on D as $F = \mathcal{T}(D)$.*

Test time

- *A (labeled) natural test set V is sampled i.i.d from (X^t, Y^t), V is sent to the attacker.*
- **(Attacker)** *Produce an unlabeled dataset U as follows:*
  1. *On input Γ (defender's adaptation algorithm), F (pretrained model), D (source training set), and V (clean test set), $\mathcal{A}$ perturbs each point $(x, y) \in V$ to (x', y) (subject to the agreed attack type), giving $\widetilde{V} = \mathcal{A}(\Gamma, F, D, V)$.*
  2. *Send $U = \widetilde{V}|_X$ (that is, the feature vectors of $\widetilde{V}$) to the defender.*
- **(Defender)** *Produce an adapted model as $\widetilde{F} = \Gamma(F, D, U)$.*

Evaluation: *Evaluate the test loss of $\widetilde{F}$ on $\widetilde{V}$, $L(\widetilde{F}, \widetilde{V}) = \frac{1}{|\widetilde{V}|}\sum_{(x',y)\in\widetilde{V}} \ell(\widetilde{F}; x', y)$. Attacker's goal is to maximize the test loss, while the defender's goal is to minimize the test loss.*

On the adversary: *While the adversary knows the adaptation algorithm, Γ may use some private randomness, such as random initialization. Since defender proceeds after the attacker to apply Γ, these private randomness is assumed not to be known by the adversary.*

On notations: *If there is no need for a pretrained model, we use the notation $\Gamma(\perp, \cdot, \cdot)$.*

Remarks. We note the following aspects of this threat model:

Source domain may differ from the target domain. However, in the case where these two domains coincide, the maximin threat model is still nontrivial. In fact, from the security perspective, both threat models are about the *same* end-to-end security goal: *Correct predictions on potentially adversarially perturbed data sampled from (X, Y).*

Time-time. This is called a *test-time* threat model since $\widetilde{F}$ is trained based on $U = \widetilde{V}|_X$, and the test error is only evaluated on $\widetilde{V}$ (the perturbed test set). This is in sharp contrast with the classic minimax threat model, where the model F is trained on D, which is independent from V and so $\widetilde{V}$.

Maximin. This is called a *maximin* threat model since the adversary *must move first* to submit the challenge set U, and the defender then moves based on it. In fact, this can be written as a maximin game $\mathbb{E}_V \left[\max_{\widetilde{V}} \min_{\widetilde{F}} \text{test_err}(\widetilde{F}, \widetilde{V}) \right]$. Different from the classic minimax threat model, the expectation is taken over the entire maximin, and the exact dependencies of $\widetilde{F}$ are described by the threat model (for example, $\widetilde{F}$ can only be built from *partial information* of $\widetilde{V}$, i.e. $\widetilde{V}|_X$). We prove in Appendix A that maximin threat model is at least *no harder* than the minimax threat model.

Hard and easy. The defender only receives *adversarially perturbed and unlabeled* data U from a target domain, which can be different from the source domain.This makes the game "hard." On the other hand, the defender is only required to classify U, without need to generalize to future data points from the target domain. This makes the game "easy."

Test size may matter. This threat model enables us to study whether *a large test set (i.e. large $|U|$) can benefit a defender for adversarial robustness*. This is in sharp contrast with the classic minimax threat model, where the attacker is granted much power, and can pick points in a one-by-one, or "online", fashion (i.e., sample a point from the nature, perturb it, and send to the defender).

Attacking a model vs. Attacking an algorithm. In the classic threat model, a model F is fixed after training, and at the test time the adversary attacks the model. However, in the maximin threat model, the adversary must attack an adaptation *algorithm* Γ.

Test-time defenses and BPDA. Various previous work have investigated *test-time defenses* where a pretrained model is fixed and there is a "preprocessing procedure" to preprocess an input before sending it to the model (in the hope of "removing" the adversarial perturbation). Several such defenses were desribed in Athalye et al. (2018), and were broken by the BPDA technique (Backward Pass Differentiable Approximation). While syntactically one can fit test-time defenses into our framework, the test-time adaptation concept discussed in this paper is significantly different, which usually allows one to completely modify a model. As we will see later in the paper, Γ can be instantiated as an unsupervised domain adaptation algorithm, such as DANN, and obtain nontrivial robustness in the maximin threat model. In these cases, it is unclear how to apply BPDA.

3.1 Restrictions and variants of the threat model

Astute readers may realize that we have intentionally left the definition of the test-time adaptation Γ to have a lot of freedom. For example, Γ can leverage the labeled training data D, which may be practically not available or computationally infeasible for test-time computations. In this section we describe variants and restrictions of the standard model above.

Definition 3 (Online version). *In the online maximin model, we assume that we will sequentially receive batches of test samples, one after another, and we allow Γ to leverage the previous samples received. We assume, for simplicity of this work however, that the batches are sampled independently from each other.*

We also consider the following possibilities of restrictions:

- **Homogeneity**. If the source and target domains equal, we call it the *homogeneous maximin model*. This setting is directly related to the classic threat model (Definition 1), where we have a single domain (X, Y), but an adversary can perturb the test instances. From a security perspective, maximin threat model gives a *relaxed* threat model, but *the same end-to-end security guarantees*. (i.e. correct predictions against adversarially perturbed input)
- **Data-obliviousness**. In this case, the adaptation algorithm Γ cannot leverage the labeled source data D. In other words, the agent can only leverage the pretrained model F and the unlabeled data set U.
  For the online maximin model, data obliviousness means that the agent cannot directly leverage the previous unlabeled samples he or she received. However, the agent can save the previous model parameters and use that.
- **Singleton**. In the standard model, we allow U to be a batch which consists of many point. In the singleton restriction, U is restricted to have only one point (for example, the test-time training algorithm by Sun et al. (2020) works in this setting).

3.2 Attacks in the maximin threat model

Algorithm 1 $\mathsf{FPA}[k, U_0]$, FIXED POINT ATTACK WITHOUT PRETRAINING

Require: A training dataset D, a natural test set V, an initial perturbed feature set U_0, and an integer parameter $k \geq 1$.

1: **for** $i = 1, 2, \ldots, k$ **do**
2: Compute $F_i = \Gamma(\perp, D, U_{i-1})$.
3: Attack F_i on V to generate adversarial examples V_i, and set $U_i = V_i|_X$.
4: **end for**
5: **return** U_k.

Algorithm 2 $\mathsf{FPA}[k, F]$, FIXED POINT ATTACK WITH PRETRAINING

Require: A training dataset D, a natural test set V, a pretrained model F, and an integer parameter $k \geq 0$.

1: Set $F_0 = F$. Attack F_0 on V to get adversarially perturbed examples V_0. Set $U_0 = V_0|_X$.
2: **for** $i = 1, 2, \ldots, k$ **do**
3: Compute $F_i = \Gamma(F, D, U_{i-1})$.
4: Attack F_i on V to get adversarially perturbed V_i. Set $U_i = V_i|_X$.
5: **end for**
6: **return** U_k.

We consider two categories of attacks in this paper:

Transfer Attacks. Transfer attacks are a common class of attacks, where we transfer attacks on one model to the target model (in our context, produced by a maximin defender strategy). In this paper we will mainly consider PGD attacks (Projected Gradient Descent attacks) with norm-based perturbations Madry et al. (2018). For example, given a model F, we use the notation $\mathsf{PGD}(F, V)$ to denote PGD attacks against F, on data V (the attack type is specified in the context). We adopt the following notations:

Notation	Meaning
T	A target model trained on the labeled target data V.
AdvT	An adversarially trained target model using the labeled target data V.
S	A source model trained on the labeled source data D.
AdvS	An adversarially trained source model using the labeled source data D.
$\mathsf{PGD}(\cdot, \cdot)$	PGD Attacks on a model and data. For example, $\mathsf{PGD}(\mathsf{AdvT}, V)$ means to use PGD attacks on the model AdvT and data V.

Fixed Point Attacks. As we discussed above, it is interesting to consider how to leverage the knowledge of an adaptation algorithm Γ (for example, if $\Gamma = \mathsf{DANN}(\mathsf{RI}, \cdot, \cdot)$) to generate *adaptive* attacks. We propose a new family of adaptive attacks, which we call the fixed point attacks (FPA). The intuition behind FPA is that since our goal is to find U on which $\Gamma(F, D, U)$ fails (thus U is a "fixed point"), we thus start with a U_0, and iteratively compute $F_i = \Gamma(F, D, U_{i-1})$, generate U_i by attacking F_i, and hope that with large number of iterations we can converge to a good $U = U_k$.

FPA comes in two flavors, depending on whether the adaptation algorithm Γ needs a pretrained model F. These two algorithms are presented in Algorithm 1 and Algorithm 2. For the no-pretraining algorithm ($\mathsf{FPA}[k, U_0]$), we will choose to initialize U_0 in different ways in order to identify the "fixed point" U. For example, one can set $U_0 = V|_X$ (the raw test features), or $U_0 = \mathsf{PGD}(\mathsf{AdvT}, V)|_X$. For the with-pretraining case, we can have $k = 0$, where $\mathsf{FPA}[0, F]$ means to attack the pretrained model without applying Γ.

4 ADVERSARIAL SEMI-SUPERVISED MINIMAX THREAT MODEL: AN INTERMEDIATE THREAT MODEL

An unsatisfying aspect of the theory thus far is that the classic minimax threat model is only defined in the homogeneous case, but the maximin threat model can be inhomogeneous, which makes the direct comparison difficult. To bridge the gap, we provide a generalization of the classic minimax

threat model that lies between the classic minimax threat model (Definition 1) and the test-time maximin threat model (Definition 2).

Definition 4 (**Adversarial semi-supervised minimax threat model**). *Fix an adversarial perturbation type (e.g. ℓ_∞ attacks with norm budget ε). We have a source domain (X^s, Y^s), and a target domain (X^t, Y^t). Both domains are clean data, and are not related to adversarial perturbations. The attacker is a pair of algorithms $(\mathcal{A}_0, \mathcal{A}_1)$, and the defender is a pair of algorithms $(\mathcal{T}, \Gamma)$, where $\mathcal{T}$ is a supervised learning algorithm, and Γ is a semi-supervised learning algorithm.*

Before game starts

- *A training set D is sampled i.i.d. from (X^s, Y^s).*
- *A semi-supervision set V is sampled i.i.d from (X^t, Y^t), V is sent to the attacker.*

Training time

- **(Defender, optional)** *Train a model F trained on D as $F = \mathcal{T}(D)$.*
- **(Attacker)** *Produce an unlabeled dataset U as follows:*
  1. *On input Γ (defender's semi-supervised learning algorithm), F (pretrained model), D (source training set), and V (clean test set), $\mathcal{A}_0$ perturbs each point $(x, y) \in V$ to (x', y) (subject to the agreed attack type), giving $\widetilde{V} = \mathcal{A}_0(\Gamma, F, D, V)$.*
  2. *Send $U = \widetilde{V}|_X$ (that is, the feature vectors of $\widetilde{V}$) to the defender.*
- **(Defender)** *Produce a final model as $\Gamma : \widetilde{F} = \Gamma(F, D, U)$.*

Test time

- *A (labeled) natural test set V' is sampled i.i.d. from (X^t, Y^t).*
- **(Attacker)** *On input $\widetilde{F}$ (defender's final model), D (source training set), and V' (clean test set), $\mathcal{A}_1$ perturbs each point $(x, y) \in V'$ to (x', y) (subject to the agreed attack type), giving $\widetilde{V'} = \mathcal{A}_1(\widetilde{F}, D, V')$.*

Evaluation: *Evaluate the test loss of $\widetilde{F}$ on $\widetilde{V'}$, $L(\widetilde{F}, \widetilde{V'}) = \frac{1}{|\widetilde{V'}|} \sum_{(x',y)\in\widetilde{V}} \ell(\widetilde{F}; x', y)$. Attacker's goal is to maximize the test loss, while the defender's goal is to minimize the test loss.*

Note that the procedure to produce $\widetilde{F}$ at the training time of this threat model is exactly the same as the procedure to produce $\widetilde{F}$ in the test-time maximin threat model. The key difference is that, once $\widetilde{F}$ is trained, we evaluate it on $\widetilde{V'}$, which is the adversarially perturbed data on *independently sampled* V' (that is, not only we want the model $\widetilde{F}$ to perform well on U, but also on unseen data from (X^t, Y^t)). This is thus inductive learning, and a minimax game. We note further the following aspects of this threat model:

Semi-supervised and adversarial. This threat model is *semi-supervised*, because the defender (learner) receives an unlabeled data set U from the target domain. This threat model is *adversarial*, because the attacker can adversarially perturb the clean semi-supervision set V to produce U.

Classic minimax model as a special case. The classic minimax threat model is a *special case* of this threat model, by putting source and target domains equal, and choosing a trivial Γ: $\Gamma(F, D, U) = F = \mathcal{T}(D)$. We list several other specialization of this threat model in Appendix C. Therefore, without loss of generality, for the rest of the paper, by *minimax model* we mean the *adversarial semi-supervised minimax threat model*.

Γ: One algorithm, two interpretations. We note that for an algorithm $\Gamma : (F, D, U) \mapsto \widetilde{F}$, one can have now two interpretations: In the maximin threat model, we interpret it as a adaptation algorithm, because we are only going to evaluate $\widetilde{F}$ on $\widetilde{V}$; in the minimax threat model, we interpret it as a semi-supervied learning algorithm, because we are going to evaluate $\widetilde{F}$ on unseen points $\widetilde{V'}$.

To conclude this section, we note the following lemma,

Lemma 1 (**Good minimax solution is also a good maximin solution**). *Suppose $\mathcal{T}^*$ is a supervised learning algorithm which trains a model $F^* = \mathcal{T}^*(D)$, where its adversarial gain in the adversarial*

semi-supervised minimax model is bounded by κ (i.e. $\mathbb{E}_{V'}[\max_{\widetilde{V}'} L(F^*, \widetilde{V})] \leq \kappa$.) Then in the maximin threat model, the adversarial gain of the strategy $(\mathcal{T}^*, \Gamma^*)$, where $\Gamma^*(F^*, D, U) = F^* = \mathcal{T}^*(D)$, is also upper bounded by κ.

5 MAXIMIN VS. MINIMAX: A DANN PERSPECTIVE

We have shown that the maximin threat model is *no harder* than the minimax threat model. (Appendix A). In fact, by Lemma 1, we know that a good model F in the minimax model would trivially induce a good defender adaptation strategy (by just outputting that model). Therefore, intuitively, the "good solutions" in the minimax threat model is a subset of the "good solutions" in the maximin threat model. How about the reverse direction? This leads us to the following:

Is there a Γ such that:

- *As a test-time adaptation algorithm in the maximin model, it provides robustness, but*
- *As a semi-supervised learning algorithm in the minimax model, the model it produces has no robustness.*

The existence of such a Γ provides a separation between the minimax and the maximin threat models, and indicates that the maximin threat model admits more good solutions. Despite theoretical interest, this question is of practical importance since these "new" solutions may possess *desirable properties that good solutions in the minimax threat model may lack*.

In this section we consider one such desirable property, as described by Goodfellow (2018) (pp.30): *The defender strategy is not to directly optimize the performance measure* (e.g., we know the attack is ℓ_∞-norm attacks, and we directly train for it). While we are not able to exhibit such a strategy theoretically, in this section we design and implement experiments to provide evidence that such a strategy indeed exists.

High-level experiment design. At the high level the experiments proceed as follows:

(1) We use the DANN algorithm with random initialization (RI), as a test-time adaptation; that is, $\mathsf{DANN}(\mathsf{RI}, \cdot, \cdot)$. There are several motivations of choosing DANN:

- DANN fits our framework as it is designed for unsupervised domain adaptation.
- DANN is however *not* designed for adversarial robustness. Thus is it a very interesting question whether DANN can provide test-time adversarial robustness against attacks (e.g. norm-based attacks) that *it is not specifically designed for*.
- DANN algorithm leverages source data D, which could benefit the maximin robustness.

(2) We generate adversarially perturbed $\widetilde{V}$ using strong transfer attacks (based on PGD attacks), and adaptive fixed point attacks, and check whether $\mathsf{DANN}(\mathsf{RI}, \cdot, \cdot)$ can provide robustness on these data.

(3) We also check whether the adapted model $\widetilde{F} = \mathsf{DANN}(\mathsf{RI}, D, U)$ is robust in the minimax threat model, that is, under direct adversarial attacks. (Strictly speaking, we need to use the adversarial semi-supervised minimax threat model defined in Definition 4.)

(4) If (2) and (3) show that $\mathsf{DANN}(\mathsf{RI}, \cdot, \cdot)$ is *significantly more robust* in the test-time maximin threat model than in the minimax model, then the experiment goal is achieved.

Datasets. (XW: Needs more details here) For the homogeneous case, we consider MNIST (i.e., for both source and target domains), and CIFAR10 (i.e., for both source and target domains). The homogeneous case represents the security problem considered in the classic threat model. For the inhomogeneous case, we consider MNIST→MNIST-M Ganin et al. (2017), and CIFAR10→CIFAR10-fog Hendrycks & Dietterich (2019). For CIFAR10-fog, all 5 levels combined and perform and 80-20 train test split, to obtain the training set of size 40000 and test dataset of size 10000. Note that we choose to simply combine all corruption levels because our main goal of this experiment is not to get best numbers for adaptation, but to understand the relative difference between different methods. (XW: I don't think we should explain too much here. maybe we should make this a footnote actually.)

Models. For MNIST, we use the original DANN architecture from Ganin et al. (2017) with slight modifications, including the addition of batch normalization and dropout layers. For CIFAR10, we use pre-activation ResNets from He et al. (2016), and for the DANN and TTT branch we use the same as Sun et al. (2020), i.e. convolution layer. Note that this architecture is slightly different from the traditional DANN architecture, which assumes vectorized feature and Fully connected DANN branch.

Maximin robustness under transfer attacks.

$\mathsf{DANN}(\mathsf{RI}, D, \cdot)$	$\mathsf{PGD}(\mathsf{AdvT}, V)$
MNIST	98%
CIFAR10	49.47%
MNIST→MNIST-M	68%
CIFAR10 → CIFAR10-fog	52.28%

Table 1: Maximin robustness of DANN under transfer Attacks

Maximin robustness under fixed point attacks

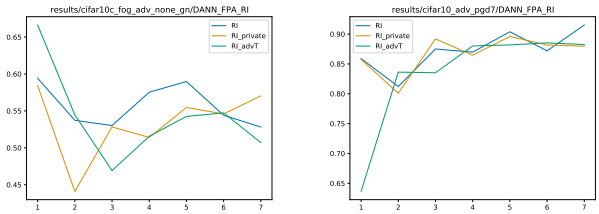

Figure 1: MNIST

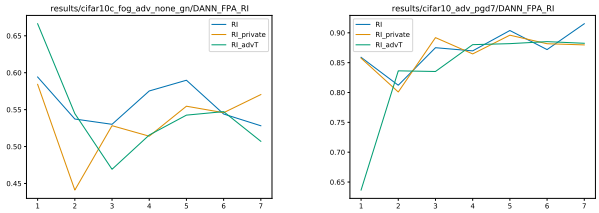

Figure 2: CIFAR10

Effect of target size.

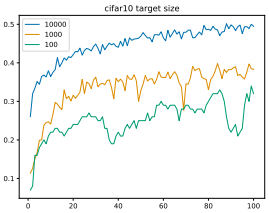

Figure 3: CIFAR10

6 TOWARDS ADVERSARIALLY ROBUST TEST-TIME ADAPTATION AND ITS IMPLICATIONS

REFERENCES

Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 274–283. PMLR, 2018.

Constantinos Daskalakis, Stratis Skoulakis, and Manolis Zampetakis. The complexity of constrained min-max optimization, 2020.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky. Domain-adversarial training of neural networks. In Gabriela Csurka (ed.), *Domain Adaptation in Computer Vision Applications*, Advances in Computer Vision and Pattern Recognition, pp. 189–209. Springer, 2017.

Shafi Goldwasser, Adam Tauman Kalai, Yael Tauman Kalai, and Omar Montasser. Beyond perturbations: Learning guarantees with arbitrary adversarial test examples. *CoRR*, abs/2007.05145, 2020. URL https://arxiv.org/abs/2007.05145.

Ian J. Goodfellow. Defense against the dark arts: An overview of adversarial example security research and future research directions. *CoRR*, abs/1806.04169, 2018. URL http://arxiv.org/abs/1806.04169.

Ian J. Goodfellow. A research agenda: Dynamic models to defend against correlated attacks. *CoRR*, abs/1903.06293, 2019. URL http://arxiv.org/abs/1903.06293.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pp. 630–645. Springer, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In Rupak Majumdar and Viktor Kuncak (eds.), *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I*, volume 10426 of *Lecture Notes in Computer Science*, pp. 97–117. Springer, 2017.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(8):1979–1993, 2019.

Zachary Nado, Shreyas Padhy, D Sculley, Alexander D'Amour, Balaji Lakshminarayanan, and Jasper Snoek. Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv preprint arXiv:2006.10963*, 2020.

Rui Shu, Hung H. Bui, Hirokazu Narui, and Stefano Ermon. A DIRT-T approach to unsupervised domain adaptation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

Yu Sun, Xiaolong Wang, Liu Zhuang, John Miller, Moritz Hardt, and Alexei A. Efros. Test-time training with self-supervision for generalization under distribution shifts. In *ICML*, 2020.

Jonathan Uesato, Jean-Baptiste Alayrac, Po-Sen Huang, Alhussein Fawzi, Robert Stanforth, and Pushmeet Kohli. Are labels required for improving adversarial robustness? In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pp. 12192–12202, 2019.

Vladimir Vapnik. *Statistical learning theory*. Wiley, 1998. ISBN 978-0-471-03003-4.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno A. Olshausen, and Trevor Darrell. Fully test-time adaptation by entropy minimization. *CoRR*, abs/2006.10726, 2020.

A MAXIMIN THREAT MODEL IS NO HARDER THAN THE MINIMAX THREAT MODEL

Lemma 2. *Let $k \geq 1$ be a natural number. For a given V, the domain of $\widetilde{V}$ is a well-defined function of V (e.g., ℓ_∞ ball around V). We have that:*

$$\mathop{\mathbb{E}}_{V\sim(X,Y)^k}\left[\max_{\widetilde{V}}\min_{\widetilde{F}}\{L(\widetilde{F},\widetilde{V})\}\right] \leq \min_{\widetilde{F}}\mathop{\mathbb{E}}_{V\sim(X,Y)^k}\left[\max_{\widetilde{V}}\{L(\widetilde{F},\widetilde{V})\}\right]$$

Proof. Let $\mathcal{F}$ be the family of models $\widetilde{F}$ we can choose from. From the maximin inequality, we have that

$$\max_{\widetilde{V}}\min_{\widetilde{F}}\{L(\widetilde{F},\widetilde{V})\} \leq \min_{\widetilde{F}}\max_{\widetilde{V}}\{L(\widetilde{F},\widetilde{V})\}$$

If we take expectation over V, we have thus

$$\mathop{\mathbb{E}}_{V}\left[\max_{\widetilde{V}}\min_{\widetilde{F}}\{L(\widetilde{F},\widetilde{V})\}\right] \leq \mathop{\mathbb{E}}_{V}\left[\min_{\widetilde{F}}\max_{\widetilde{V}}\{L(\widetilde{F},\widetilde{V})\}\right]$$

Note that

$$\mathop{\mathbb{E}}_{V}\left[\min_{\widetilde{F}}\max_{\widetilde{V}}\{L(\widetilde{F},\widetilde{V})\}\right] \leq \min_{\widetilde{F}}\mathop{\mathbb{E}}_{V}\left[\max_{\widetilde{V}}\{L(\widetilde{F},\widetilde{V})\}\right]$$

This completes the proof. □

B PROOF OF LEMMA 1

In the maximin threat model, let us take the strategy $(\mathcal{T}^*, \Gamma^*)$ as defined. Therefore the maximin adversarial gain is

$$\mathop{\mathbb{E}}_{V}\left[\max_{\widetilde{V}} L(\Gamma^*(\mathcal{T}^*(D), D, \widetilde{V}|_X), \widetilde{V})\right] = \mathop{\mathbb{E}}_{V}\left[\max_{\widetilde{V}} L(\mathcal{T}^*(D), \widetilde{V})\right] = \mathop{\mathbb{E}}_{V}\left[\max_{\widetilde{V}} L(F^*, \widetilde{V})\right]$$

which, by assumption, is bounded by κ. □

C SPECIALIZATION OF THE ADVERSARIAL SEMI-SUPERVISED MINIMAX THREAT MODEL

Adversarial training considered in Miyato et al. (2019); Shu et al. (2018). The setting becomes a special case of our general minimax threat model by:

- Instantiating $\mathcal{A}_0$ as a trivial adversary that does not do any perturbation. That is the defender receives a set of clean set U of unlabeled features.
- The defender adaptation stategy does adversarial training on the unlabled data, given by

$$d(p(y|x), p(y|x + r^*, \theta))$$
$$\text{where } r^* = \operatorname*{argmax}_{r:\|r\|\leq\varepsilon} d(p(y|x), p(y|x + r, \theta))$$

Unsupervised Adversarial Training Uesato et al. (2019) This is similar to the above except that the source and target domains are equal.

D MORE EXPERIMENTS

D.1 BASELINES: ACCURACY OF ADVERSARIALLY TRAINED MODELS.

Data	Attack distance	Training PGD steps ($\mathcal{T}$)	Test PGD steps ($\mathcal{A}$)	Nat. test acc	Adv. test acc
CIFAR10	8/255	7	20	78.31	46.39
CIFAR10c-fog	8/255	7	20	71.39	29.1
MNIST	0.3	40	100	98.79	98.14
MNIST-M	8/255	40	100	94.33	79.5

Table 2: Accuracy of adversarially trained models (in the minimax threat model).

Table 2 gives the test accuracy of adversarially trained models with a PGD adversary (XW: if we have test PGD steps, that means we are doing adv. test accuracy, why do we need to report nat. accuracy?)